



Theory and Methods of Lightfield Photography

SIGGRAPH 2009

Todor Georgiev
Adobe Systems
tgeorgie@adobe.com

Andrew Lumsdaine
Indiana University
lums@cs.indiana.edu

Web Page

<http://www.tgeorgiev.net/Asia2009/>

Course Description

Computational photography is based on capturing and processing discrete representations of all the light rays in the 3D space of a scene. Compared to conventional photography, which captures 2D images, computational photography captures the entire 4D “lightfield,” i.e., the full 4D radiance. To multiplex the 4D radiance onto conventional 2D sensors, light-field photography demands sophisticated optics and imaging technology. At the same time, 2D image creation is based on creating 2D projections of the 4D radiance.

This course presents light-field analysis in a rigorous, yet accessible, mathematical way, which often leads to surprisingly direct solutions. The mathematical foundations will be used to develop computational methods for lightfield processing and image rendering, including digital refocusing and perspective viewing. While emphasizing theoretical understanding, we also explain approaches and engineering solutions to practical problems in computational photography.

As part of the course, we will demonstrate a number of working light-field cameras that implement different methods for radiance capture, including the microlens approach of Lippmann and the plenoptic camera; the focused plenoptic camera, the Adobe lens-prism camera; and a new camera using a “mosquito net” mask. Various computational techniques for processing captured light-fields will also be demonstrated, including the focused plenoptic camera and real-time radiance rendering.

Prerequisites

This course is intended for anyone interested in learning about lightfield photography. The prerequisites are a basic understanding of ray optics. The course is of intermediate difficulty.

Course Syllabus

Background and Motivation

We open the course by discussing some of the fundamental limitations with conventional photography and present some motivating examples of how lightfield photography (radiance photography) can overcome these limitations.

Radiance Theory and Modeling

The theory and practice of radiance photography requires a precise mathematical model of the radiance function and of the basic transformations that can be applied to it.

Ray Transforms

We begin the theoretical portion of the course by presenting basic ray optics and ray transformations, cast in the language of matrix operations in phase space. This portion of the tutorial will cover:

- Position / Direction parameterization

- Transport through space
- Lens transformation
- Transformations in phase space
- Composition of Optical Elements
- Principle Planes

Radiance

With the machinery of ray transforms in hand, we can characterize how optical elements will transform radiance.

- Mathematical properties of radiance
- Conservation of volume
- Conservation of radiance
- Transformation by optical elements
- Image rendering

Capturing Radiance with Radiance Cameras

Although radiance is a 4-dimensional quantity, to capture it, we still must use 2-dimensional sensors. In this portion of the tutorial we discuss how cameras can be constructed to multiplex 4-dimensional radiance data as a 2-dimensional image. Beginning with basic camera models, we will develop and analyze

- Pinhole camera
- “2F” camera
- Traditional 2D camera
- Ives’ camera
- Lippmann’s camera
- Camera arrays

Radiance in the Frequency Domain

Analyzing radiance in the frequency domain provides some interesting new insights into radiance cameras as well as some surprising new types of cameras. In this portion of the course, we will discuss

- Fourier transform of radiance
- Fourier transform of radiance transforms
- Cameras of Ives and Lippmann
- MERL heterodying cameras

The Focused Plenoptic Camera

Recently, a new type of plenoptic camera has been developed that provides significantly improved spatial resolution when compared to the traditional approach. In this portion of the course, we develop an analyze the focused plenoptic camera (Plenoptic 2.0), focusing on the following topics:

- Adelson's plenoptic camera
- Focused plenoptic camera
- Comparison and contract of the two plenoptic camera approaches
- Comparison of the microimages
- Sampling in phase space

Hands-On with Radiance Cameras

A number of different working radiance cameras will be demonstrated and different particular approaches to radiance capture will be highlighted. Tutorial participants will have hands-on with the following radiance cameras:

- Microlens approach of Lippmann (showing working microlens arrays)
- Plenoptic camera (demonstrating plenoptic camera in action)
- MERL mask enhanced cameras (showing masks and coding approaches)
- Adobe lens-prism camera (showing the lenses)
- "Mosquito net" mask camera

Computational Methods for Radiance

Radiance photography has been made practical by the availability of computational techniques that can perform 2D image rendering from the 4D radiance function. The following computational issues will be discussed during this portion of the tutorial:

- Sensors, pixels, digital image representations
- Image rendering
- Space multiplexing
- Frequency multiplexing ("heterodyning")
- Fourier-slice refocusing
- Methods for Plenoptic 2.0
- Efficient (real-time) implementation using GPU Hardware

Timeline

- 8:30-8:40 Background and Motivation
- 8:40-10:25 Radiance Theory and Modeling
 - 8:40-9:05 Ray Transforms
 - 9:05-9:20 Radiance
 - 9:20-9:35 Capturing Radiance with Cameras
 - 9:35-9:50 Radiance in the Frequency Domain

- 9:50-10:25 The Focused Plenoptic Camera
- 10:25-10:30 Hardware Demonstration
- 10:30-10:45 Break and Hands On with Radiance Cameras
- 10:45-12:15 Computational Methods for Radiance
 - 10:45-11:15 Methods for Plenoptic 1.0
 - 11:15-11:30 Fourier Slice Refocusing
 - 11:30-11:50 Methods for Plenoptic 2.0
 - 11:50-12:10 Efficient Implementation with GPU Hardware
- 12:10-12:15 Wrap Up

About the Presenters

Todor Georgiev is a researcher at Adobe Systems, working closely with the Photoshop group. Having extensive background in theoretical physics, he concentrates on applications of mathematical methods taken from physics to image processing, graphics, and vision. He is the author of the Healing Brush tool in Photoshop (2002), the method better known as Poisson image editing. He has published several articles on applications of the mathematics of covariant derivatives in image processing and vision. He is working on a wide range of theoretical and practical ideas in optics, light field cameras and capture/manipulations of the optical field. His recent work concentrates on radiance camera designs. He has a number of papers and patents in these and related areas.

Andrew Lumsdaine received the PhD degree in electrical engineering and computer science from the Massachusetts Institute of Technology in 1992. He is presently a professor of computer science at Indiana University, where he is also the director of the Open Systems Laboratory. His research interests include computational science and engineering, parallel and distributed computing, mathematical software, numerical analysis, and radiance photography. He is a member of the IEEE, the IEEE Computer Society, the ACM, and SIAM.



Lightfield Photography

Todor Georgiev
Adobe Systems

Andrew Lumsdaine
Indiana University



Background and Motivation



What Is Wrong with This Image?





What Is Wrong? It's Just a Picture!





What is Wrong with Pictures?



- ▶ *The current most perfect photographic print only shows one aspect of reality; it reduces to a single image fixed on a plane, similar to a drawing or a hand-drawn painting. The direct view of reality offers, as we know, infinitely more variety. We see objects in space, in their true size, and with depth, not in a plane.*



Can We Create More than Pictures?



- ▶ *Can we request that Photography renders the full variety offered by the direct observation of objects? Is it possible to create a photographic print in such a manner that it represents the exterior world framed, in appearance, between the boundaries of the print, as if those boundaries were that of a window opened on reality.*



Different views





Different views





Different views





Different views





Different views





Different views





Different views





Different views



Change Aperture / All-In-Focus



Rodin Radiance Capture



Change Viewpoint



Change Focus



Change Focus





Radiance (aka Lightfield) Photography

- ▶ A picture is a rendering of the light rays in a scene
- ▶ Determined by lenses, aperture, viewpoint, etc.
- ▶ Radiance (lightfield) photography records the rays
 - ▶ Rays can be reproduced to render “the full variety offered by the direct observation of objects”
 - ▶ We can also synthesize arbitrary pictures
- ▶ Idea is over 100 years old (Integral photographs)
 - ▶ Technology now exists to make radiance photography practical



Course Outline

1. Background and Motivation
2. Ray Transforms
3. Radiance
4. Capturing Radiance with Cameras
5. Radiance in the Frequency Domain
6. The Focused Plenoptic Camera (Plenoptic 2.0)
7. Break and Hands-On with Plenoptic Cameras
8. Computational Methods for Radiance
9. Fourier Slice Refocusing
10. Efficient Implementation with GPU
11. Literature



Radiance Theory and Modeling

The laws of geometric optics and radiance transforms



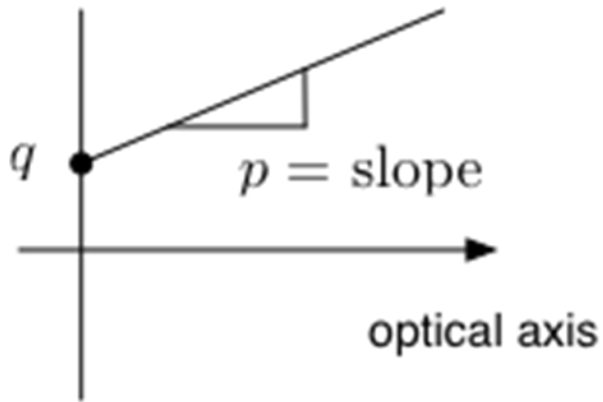
Ray Transforms

The main laws of geometric optics

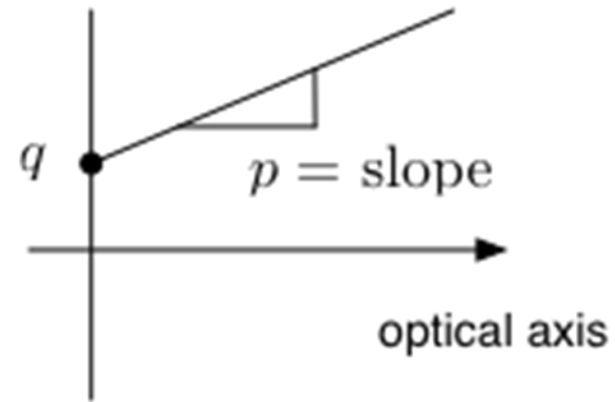


Two Parameterizations of Rays

Two-Plane



Location-Angle





Transport Through Space

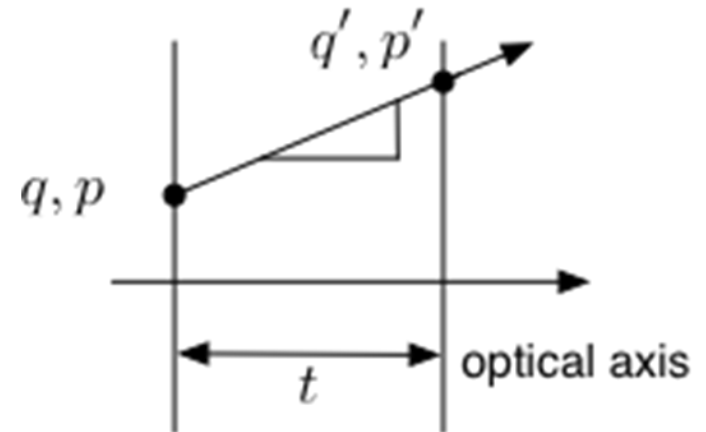
- ▶ Ray travels distance t through space
- ▶ q and p are transformed to q' and p' :

$$q' = q + tp$$

$$p' = p$$

- ▶ In matrix notation:

$$\begin{bmatrix} q' \\ p' \end{bmatrix} = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} q \\ p \end{bmatrix} = T \begin{bmatrix} q \\ p \end{bmatrix}$$



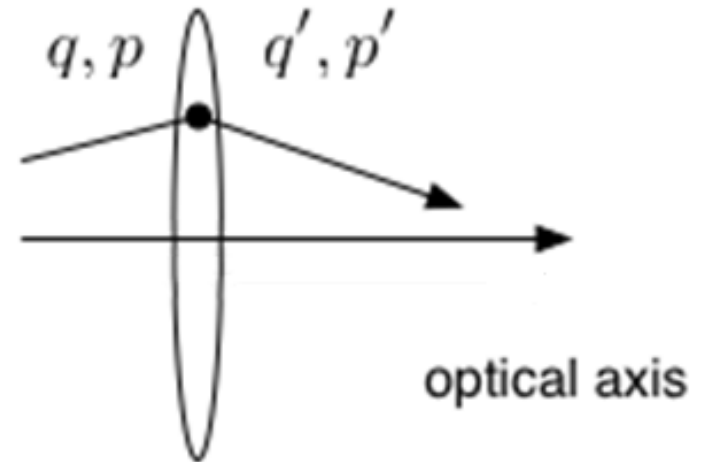


Lens Transformation

- ▶ Ray is refracted at a thin lens
- ▶ “The further from center, the more refraction”:

$$q' = q$$

$$p' = p - \frac{1}{f}q$$

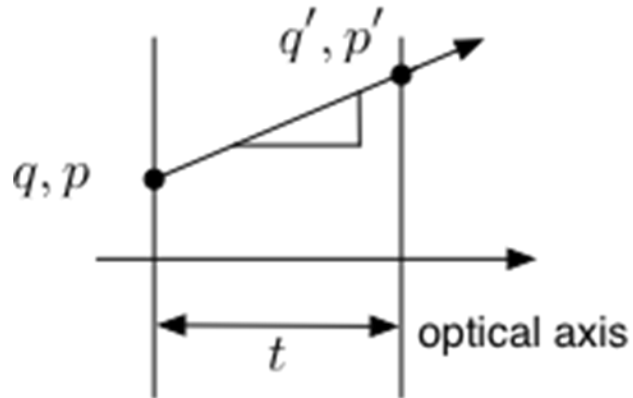


$$\begin{bmatrix} q' \\ p' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix} \begin{bmatrix} q \\ p \end{bmatrix} = L \begin{bmatrix} q \\ p \end{bmatrix}$$



Summary: Two Primary Optical Transforms

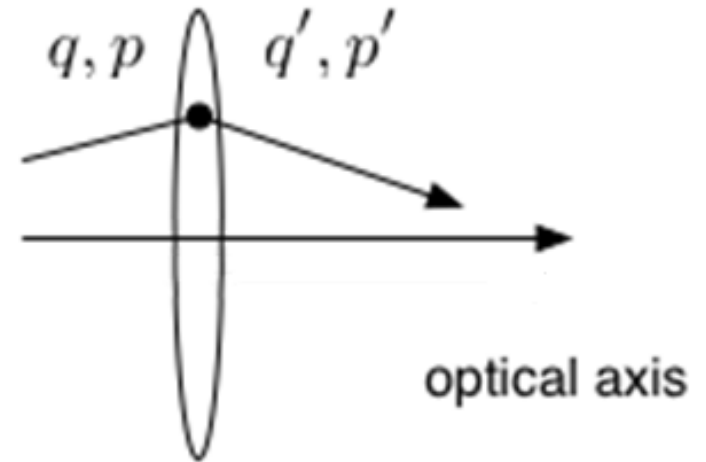
Transport



$$\begin{bmatrix} q' \\ p' \end{bmatrix} = T \begin{bmatrix} q \\ p \end{bmatrix}$$

$$T = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix}$$

Lens



$$\begin{bmatrix} q' \\ p' \end{bmatrix} = L \begin{bmatrix} q \\ p \end{bmatrix}$$

$$L = \begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix}$$

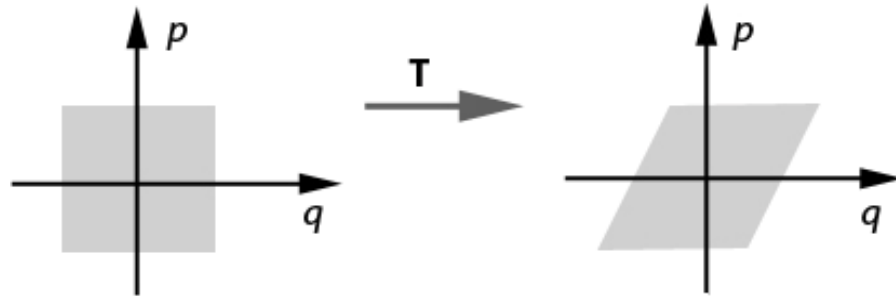
Phase Space

- ▶ This is simply the (q, p) space of rays. It is a 4D vector space with zero vector the optical axis.
- ▶ Each ray is a 4D point (a vector) in that space.
- ▶ Any optical device, like a microscope or a telescope, is a matrix that transforms an incoming ray into an outgoing ray.
- ▶ This matrix can be computed as a product of the optical elements that make up the device.

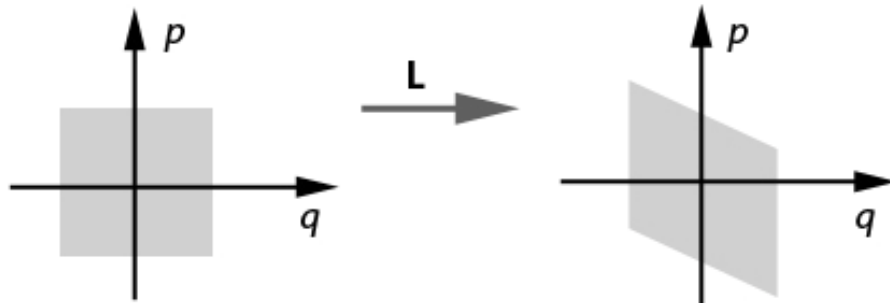


Transformations in Phase Space

Space transport



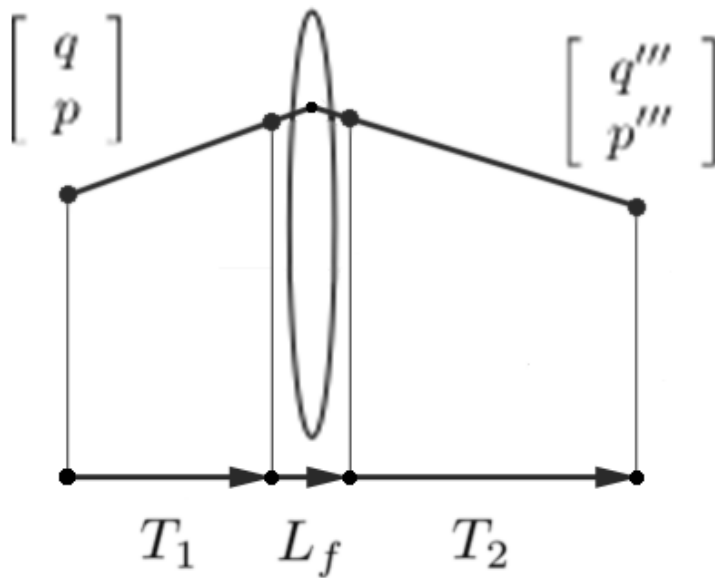
Lens refraction





Composition of Optical Elements

- ▶ Transformations corresponding to compositions of optical elements are determined by the constituent transformations

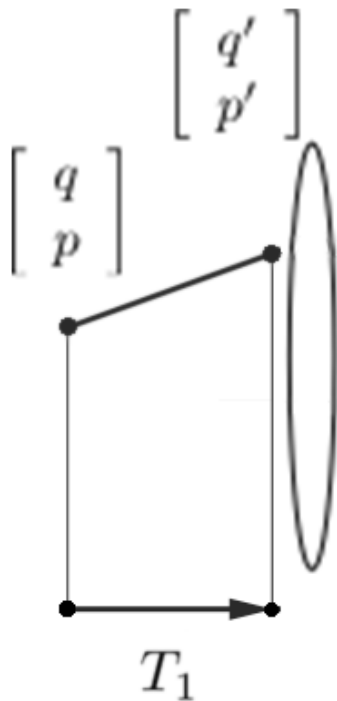


- ▶ Consider a system with transport T_1 , lens L_f and transport T_2
- ▶ What is $\begin{bmatrix} q''' \\ p''' \end{bmatrix}$ in terms of $\begin{bmatrix} q \\ p \end{bmatrix}$?



Composition of Optical Elements

- ▶ Consider one element at a time
- ▶ What is $\begin{bmatrix} q' \\ p' \end{bmatrix}$ in terms of $\begin{bmatrix} q \\ p \end{bmatrix}$?



- ▶ Transport by T_1

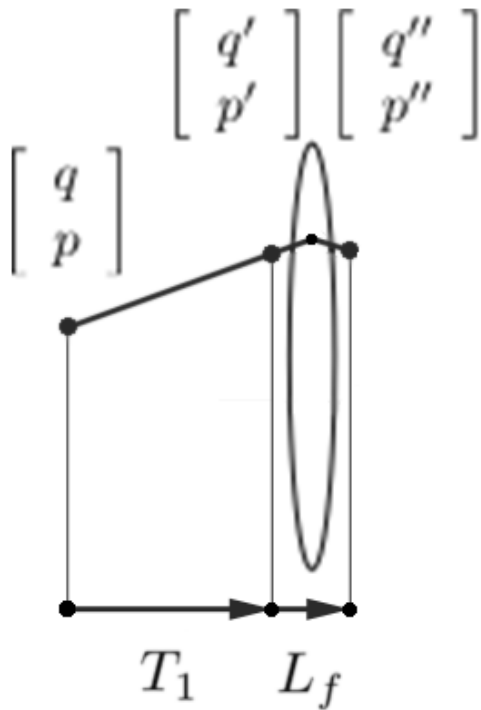
$$\begin{bmatrix} q' \\ p' \end{bmatrix} = T_1 \begin{bmatrix} q \\ p \end{bmatrix}$$

$$T_1 = \begin{bmatrix} 1 & t_1 \\ 0 & 1 \end{bmatrix}$$



Composition of Optical Elements

- ▶ Consider one element at a time
- ▶ What is $\begin{bmatrix} q'' \\ p'' \end{bmatrix}$ in terms of $\begin{bmatrix} q \\ p \end{bmatrix}$?



- ▶ Lens transform by L_f

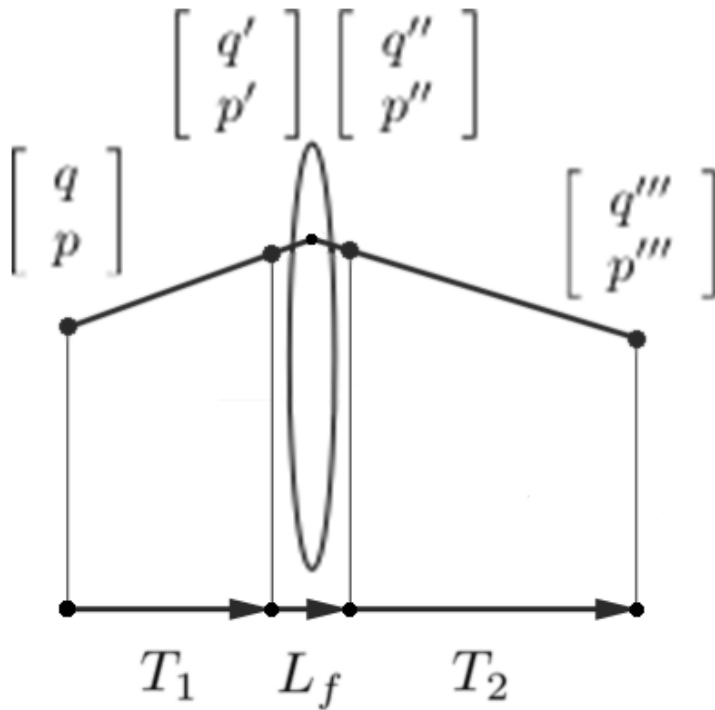
$$\begin{bmatrix} q'' \\ p'' \end{bmatrix} = L_f \begin{bmatrix} q' \\ p' \end{bmatrix}$$

- ▶ Substitute for $\begin{bmatrix} q' \\ p' \end{bmatrix}$

$$\begin{bmatrix} q'' \\ p'' \end{bmatrix} = L_f T_1 \begin{bmatrix} q \\ p \end{bmatrix}$$

Composition of Optical Elements

- ▶ Consider one element at a time
- ▶ What is $\begin{bmatrix} q''' \\ p''' \end{bmatrix}$ in terms of $\begin{bmatrix} q \\ p \end{bmatrix}$?



- ▶ Transport by $\begin{bmatrix} q' \\ p' \end{bmatrix} = T_1 \begin{bmatrix} q \\ p \end{bmatrix}$

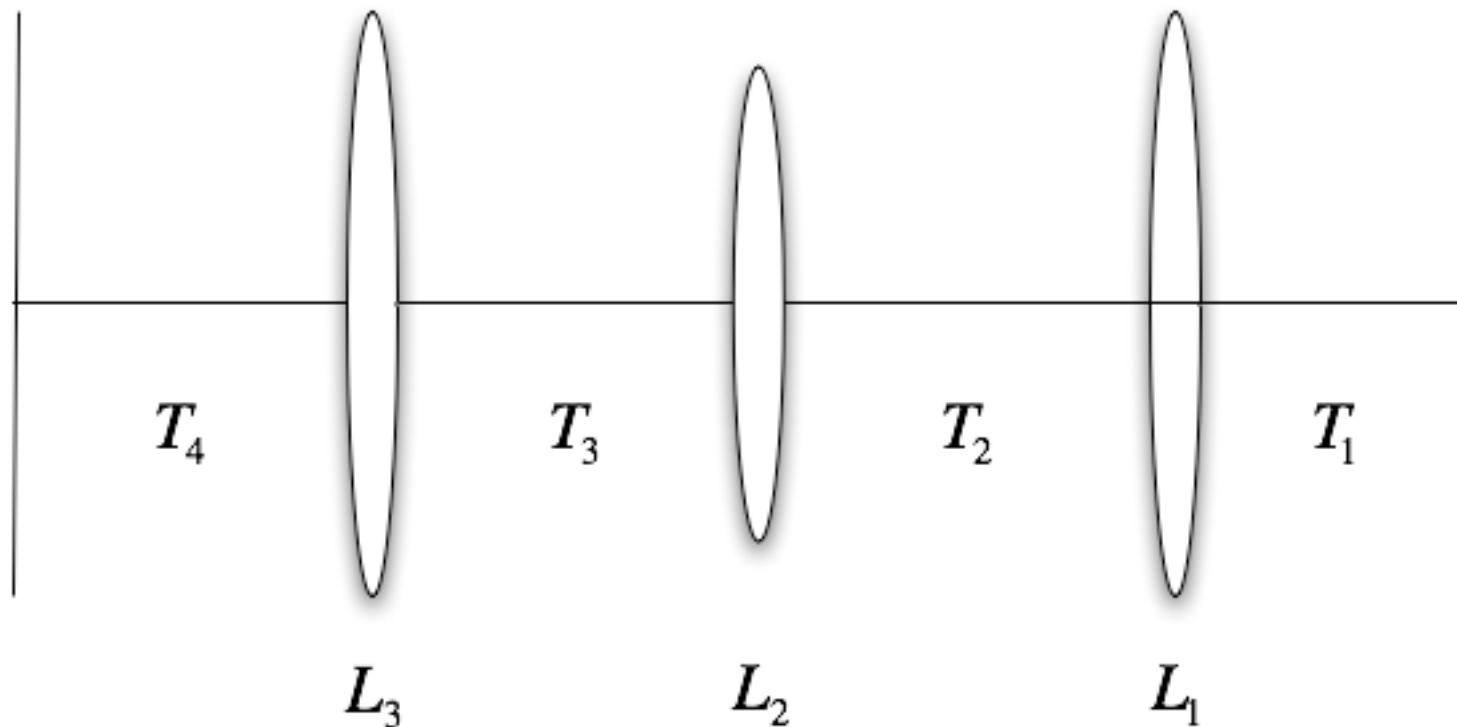
$$\begin{bmatrix} q''' \\ p''' \end{bmatrix} = T_2 \begin{bmatrix} q'' \\ p'' \end{bmatrix}$$

- ▶ Substitute for $\begin{bmatrix} q'' \\ p'' \end{bmatrix}$

$$\begin{bmatrix} q''' \\ p''' \end{bmatrix} = T_2 L_f T_1 \begin{bmatrix} q \\ p \end{bmatrix}$$

In-Class Exercise

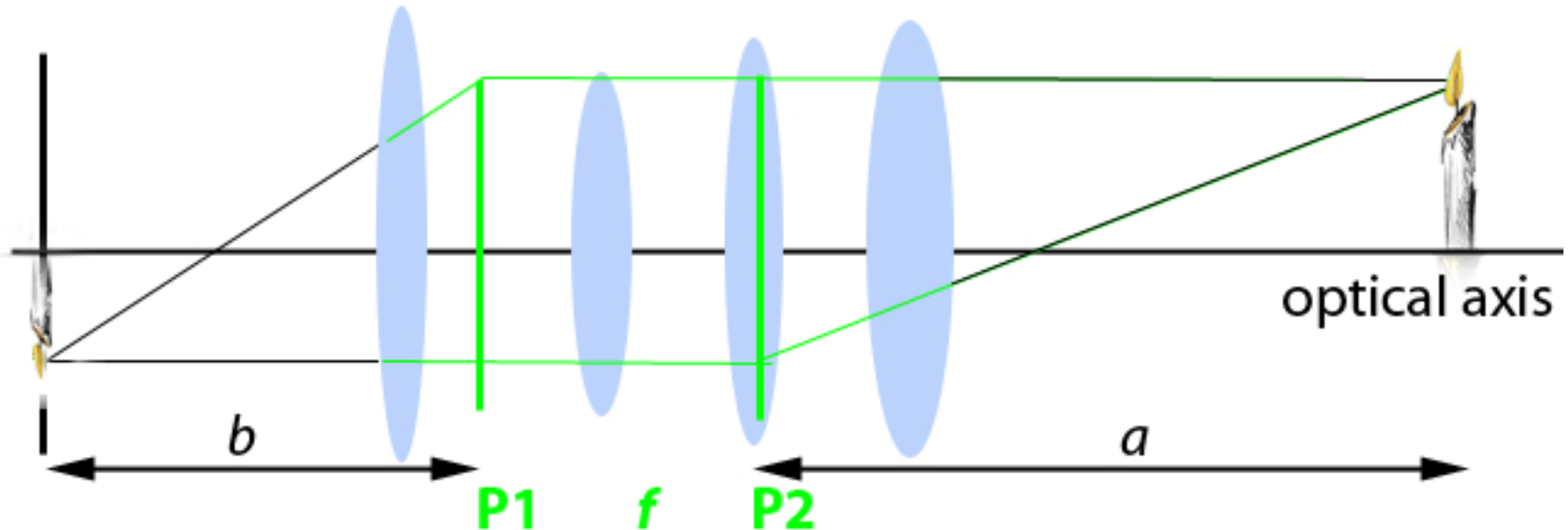
- ▶ Three-lens system
- ▶ Composition: $A = T_4 L_3 T_3 L_2 T_2 L_1 T_1$





Principal Planes

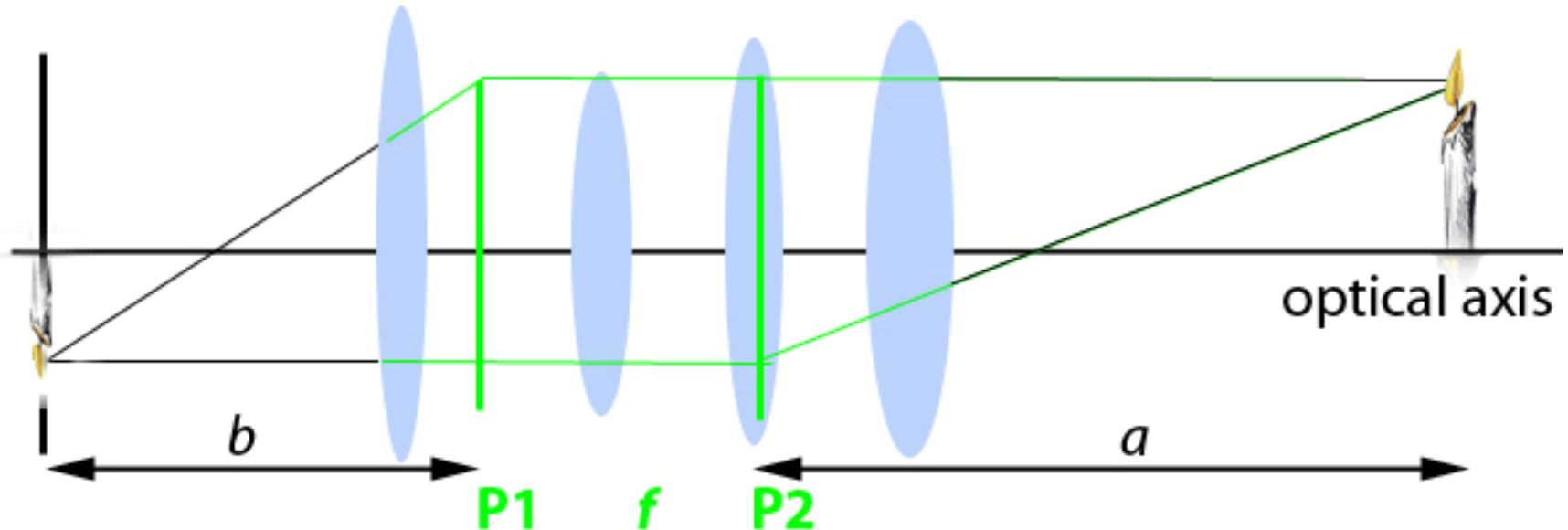
- ▶ Gauss discovered that the matrix for **any** optical transform can be written as a product of some appropriate translation, lens, and translation again.
- ▶ Often expressed as “*principal planes*” (green):





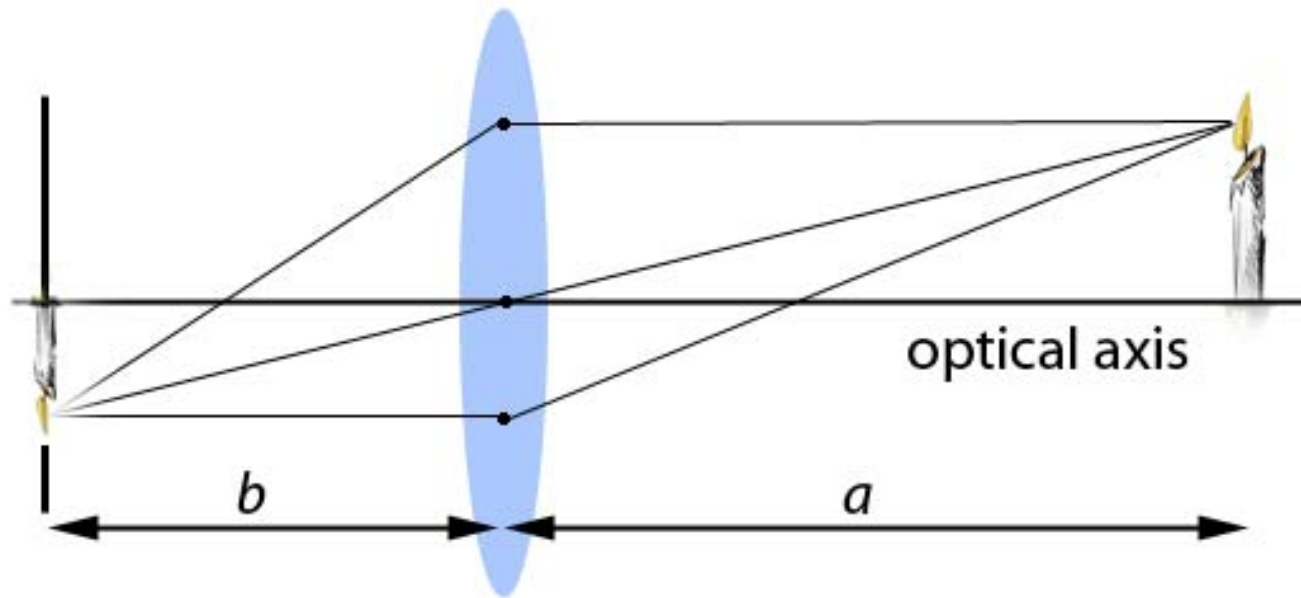
Principal Planes

- ▶ No constraint is placed on the position of the principal planes of the focal length; no travel between principal planes.





Traditional Camera

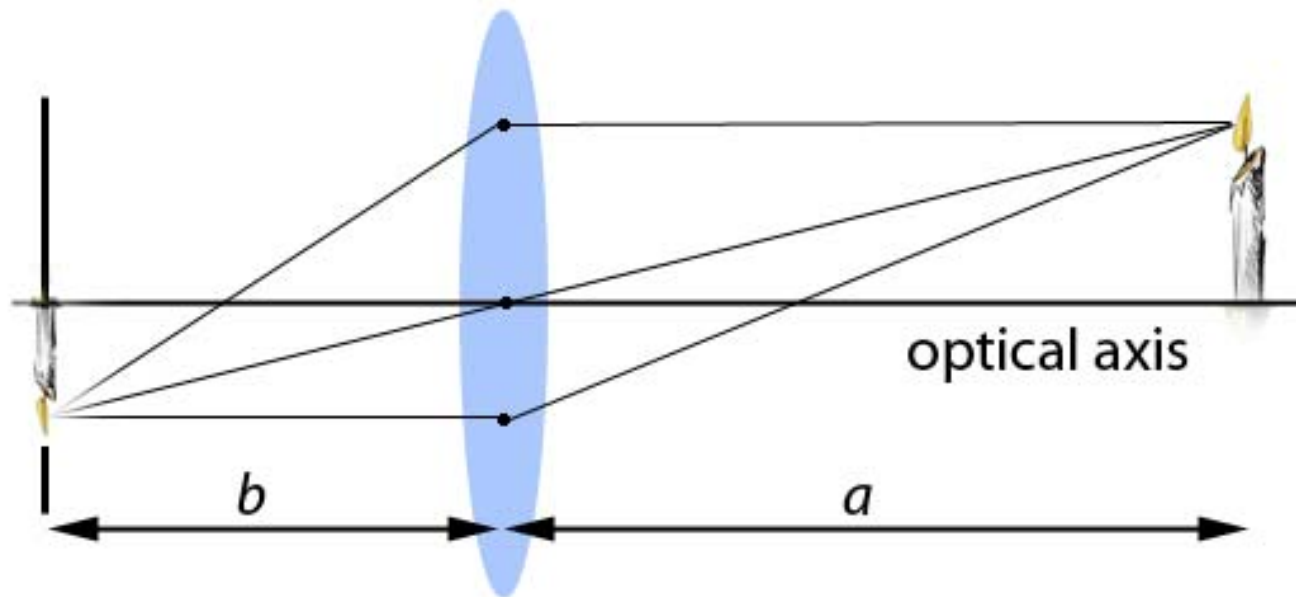


Transfer matrix:

$$A = \begin{bmatrix} 1 & b \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix} \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix}$$



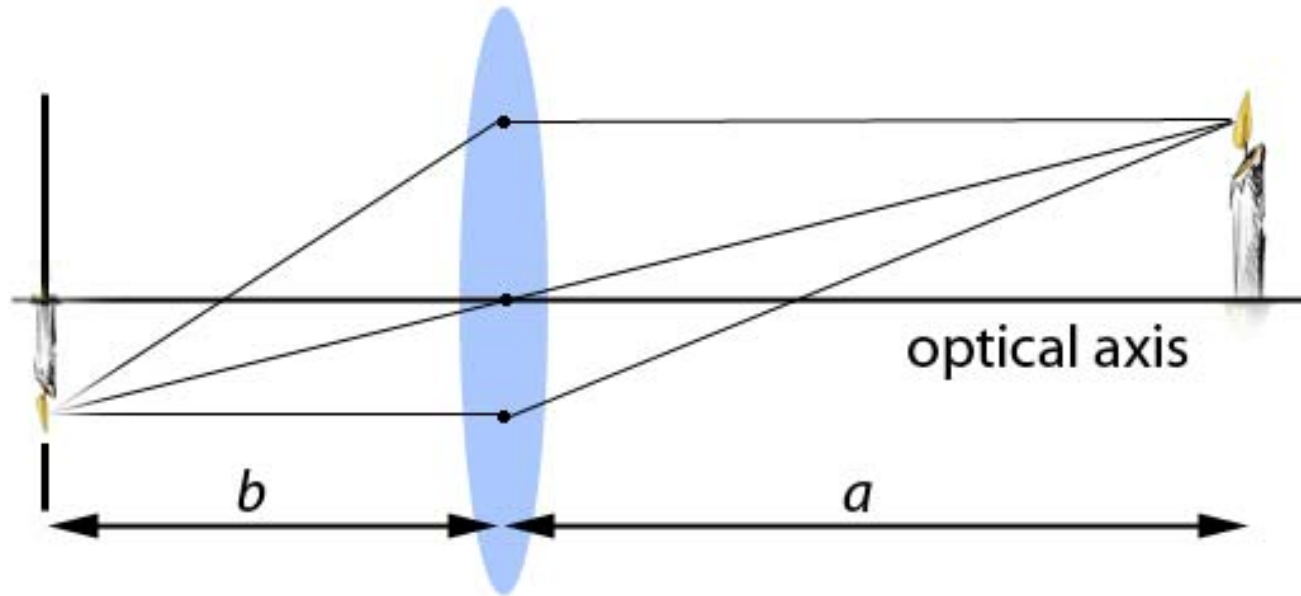
Traditional Camera



$$\begin{aligned} A &= \begin{bmatrix} 1 & b \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix} \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 1 & b \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & a \\ -\frac{1}{f} & 1 - \frac{a}{f} \end{bmatrix} = \begin{bmatrix} 1 - \frac{b}{f} & ab \left(\frac{1}{a} + \frac{1}{b} - \frac{1}{f} \right) \\ -\frac{1}{f} & 1 - \frac{a}{f} \end{bmatrix} \end{aligned}$$



Traditional Camera

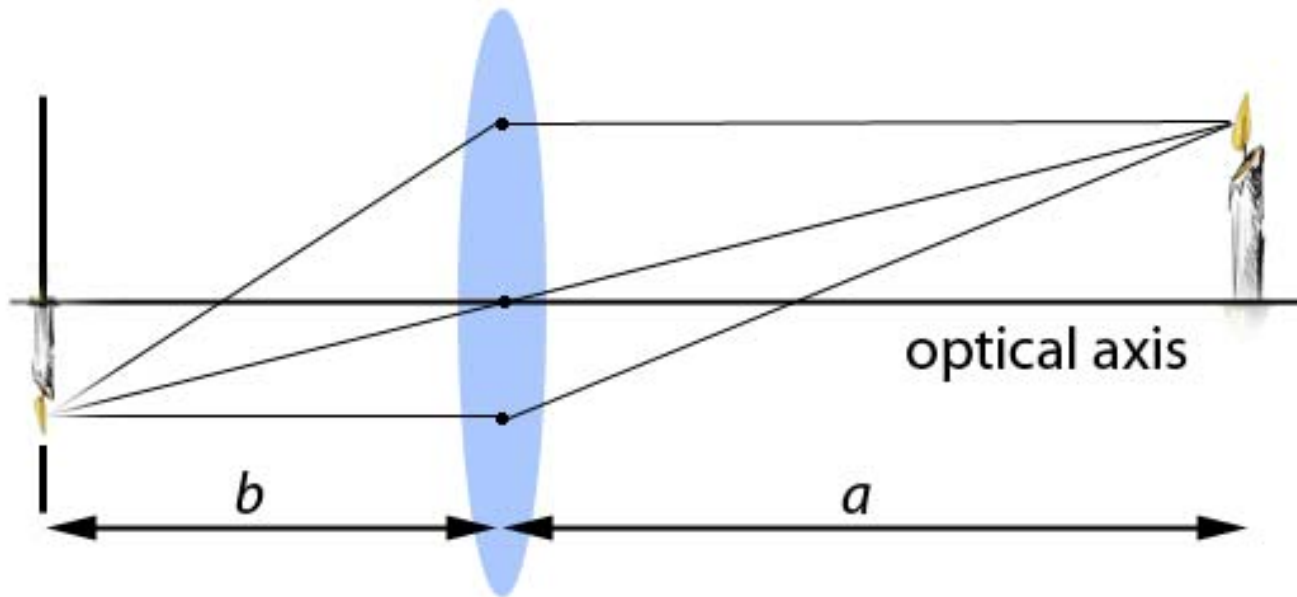


How do we focus?

$$A = \begin{bmatrix} 1 - \frac{b}{f} & ab \left(\frac{1}{a} + \frac{1}{b} - \frac{1}{f} \right) \\ -\frac{1}{f} & 1 - \frac{a}{f} \end{bmatrix}$$



Traditional Camera

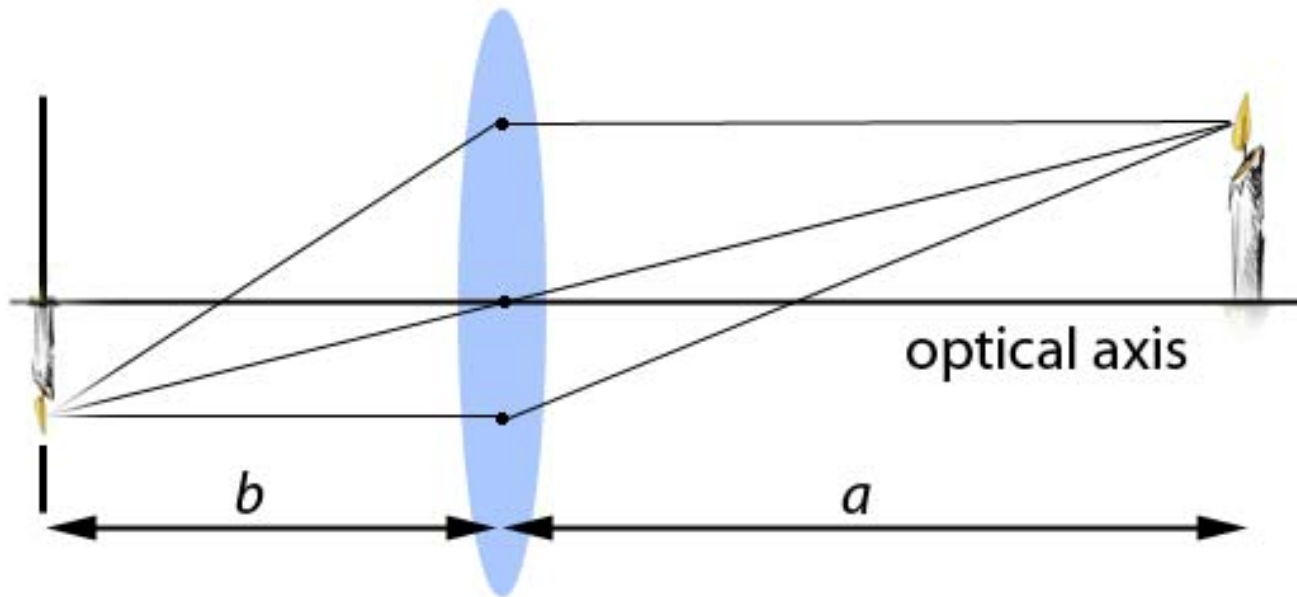


How do we focus?

Make top-right
element to be zero

$$A = \begin{bmatrix} 1 - \frac{b}{f} & ab \left(\frac{1}{a} + \frac{1}{b} - \frac{1}{f} \right) \\ -\frac{1}{f} & 1 - \frac{a}{f} \end{bmatrix}$$
$$= \begin{bmatrix} 1 - \frac{b}{f} & 0 \\ -\frac{1}{f} & 1 - \frac{a}{f} \end{bmatrix}$$

Traditional Camera



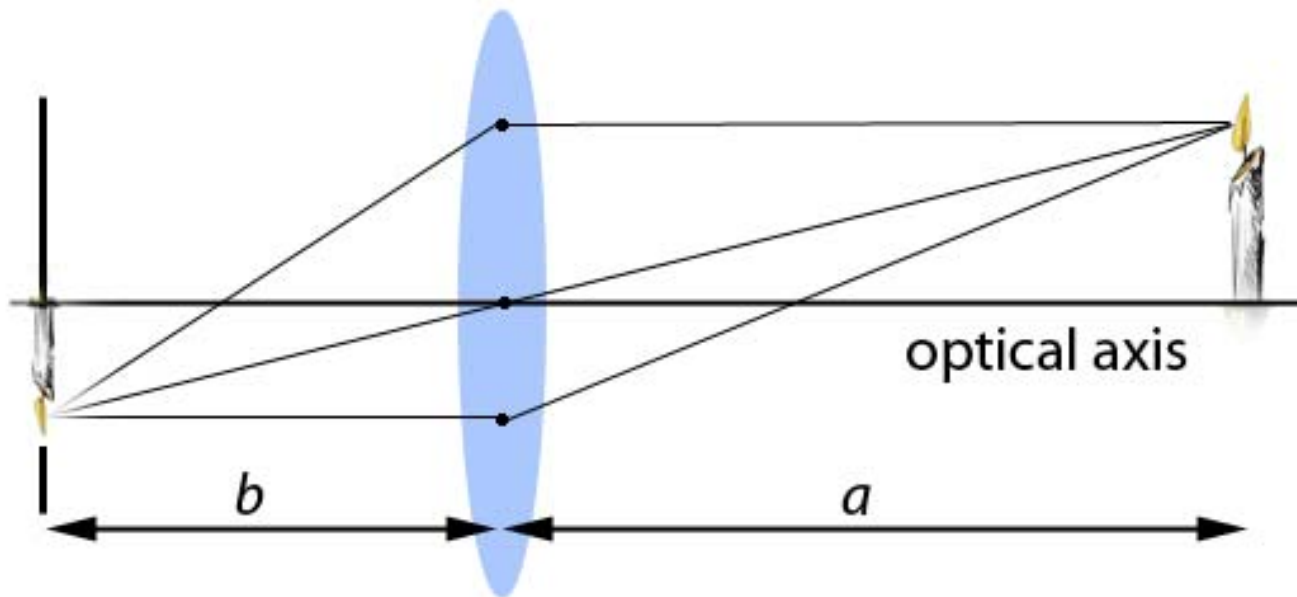
We enforce this condition by:

$$\frac{1}{a} + \frac{1}{b} - \frac{1}{f} = 0$$

$$A = \begin{bmatrix} 1 - \frac{b}{f} & ab \left(\frac{1}{a} + \frac{1}{b} - \frac{1}{f} \right) \\ -\frac{1}{f} & 1 - \frac{a}{f} \end{bmatrix}$$
$$= \begin{bmatrix} 1 - \frac{b}{f} & 0 \\ -\frac{1}{f} & 1 - \frac{a}{f} \end{bmatrix}$$



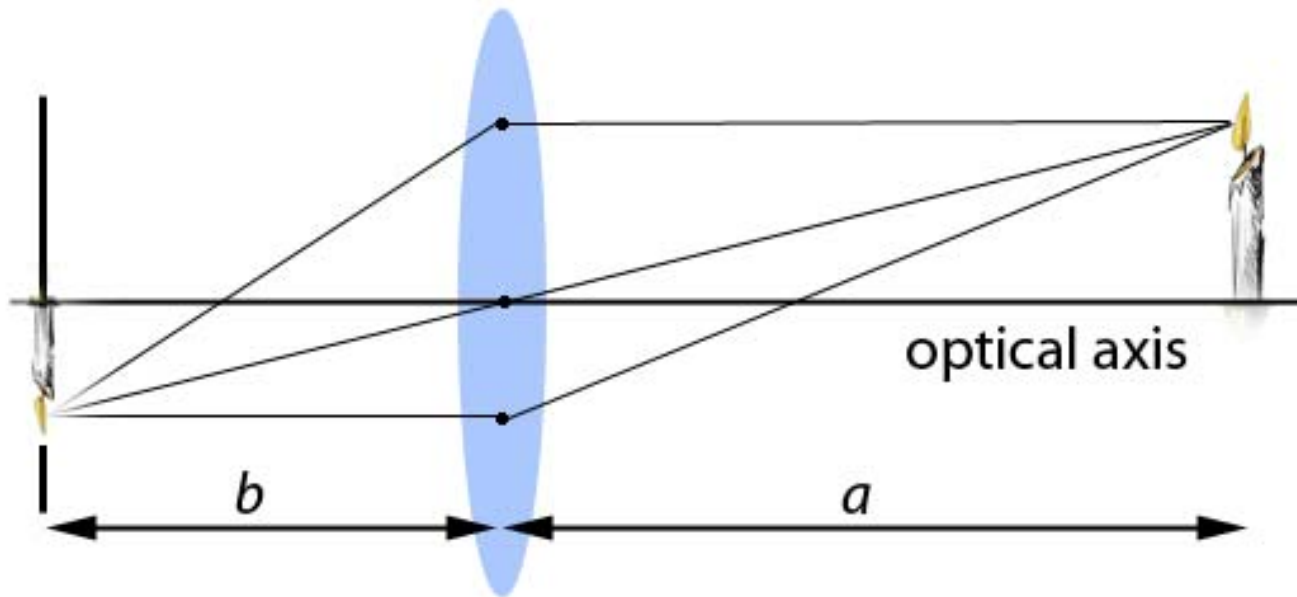
Traditional Camera



We have *derived* the lens equation: $\frac{1}{a} + \frac{1}{b} = \frac{1}{f}$

$$A = \begin{bmatrix} -\frac{b}{a} & 0 \\ -\frac{1}{f} & -\frac{a}{b} \end{bmatrix}$$

In-Class Exercise



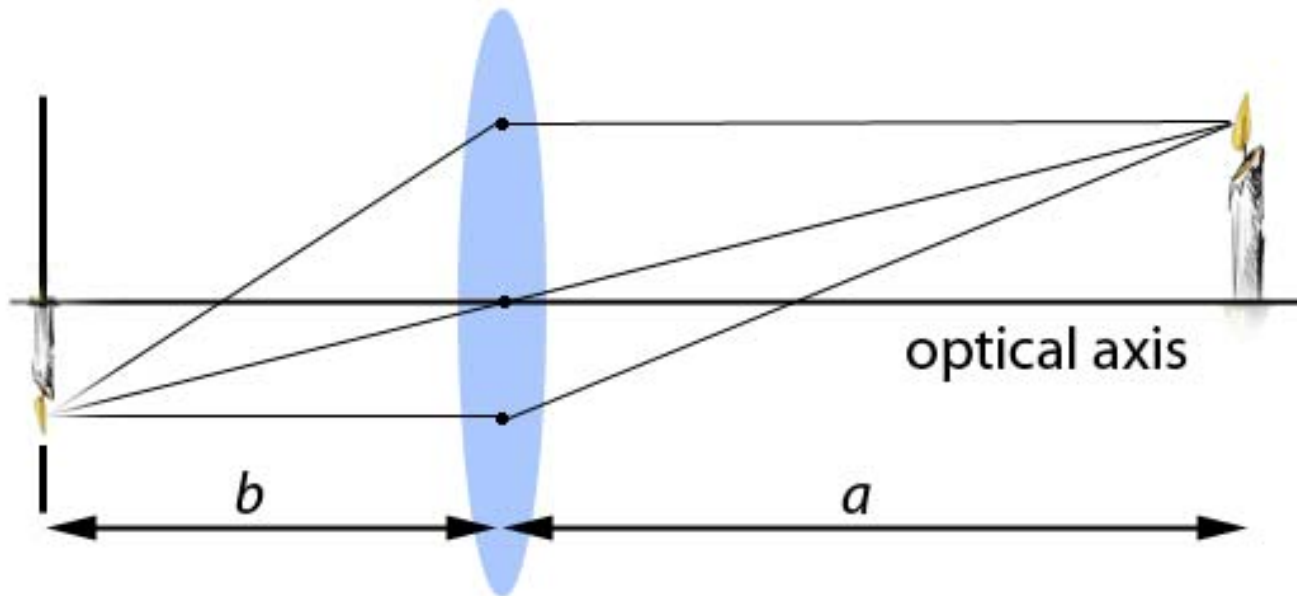
What is $\det(A)$?

Answer: $\det(A) = 1$

$$A = \begin{bmatrix} 1 - \frac{b}{f} & ab \left(\frac{1}{a} + \frac{1}{b} - \frac{1}{f} \right) \\ -\frac{1}{f} & 1 - \frac{a}{f} \end{bmatrix}$$

$$\det(A) = a_{11}a_{22} - a_{12}a_{21}$$

In-Class Exercise



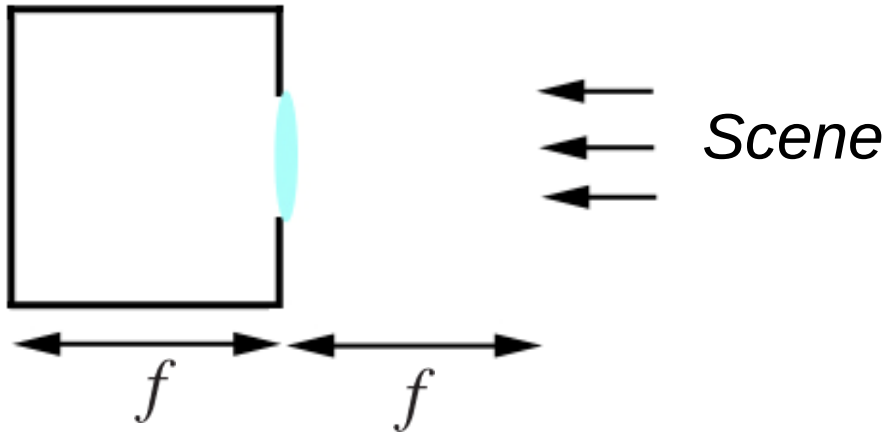
What is $\det(A)$?

$$\det(A) = 1$$

$$A = \begin{bmatrix} -\frac{b}{a} & 0 \\ -\frac{1}{f} & -\frac{a}{b} \end{bmatrix}$$

“2F” Camera

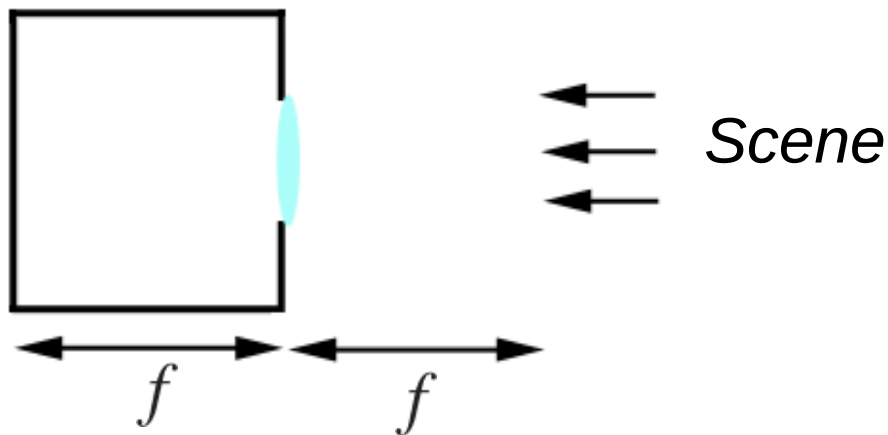
- ▶ Three optical elements: space, lens, space



- ▶ Transformation: $A = T_f L_f T_f$

In-Class Exercise

$$A = \begin{bmatrix} 1 & f \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix} \begin{bmatrix} 1 & f \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & f \\ -\frac{1}{f} & 0 \end{bmatrix}$$



What is $\det(A)$?

Again we compute $\det(A) = 1$

In-Class Exercise

- ▶ In two different cases (conventional and “2F” camera) we get the same result: $\det(A) = 1$
- ▶ Is that always the case?
- ▶ Hint: Every optical system is a composition of L and T , which both have $\det = 1$
- ▶ And the determinant of a product is the product of the determinants.
- ▶ This is an important physical property.

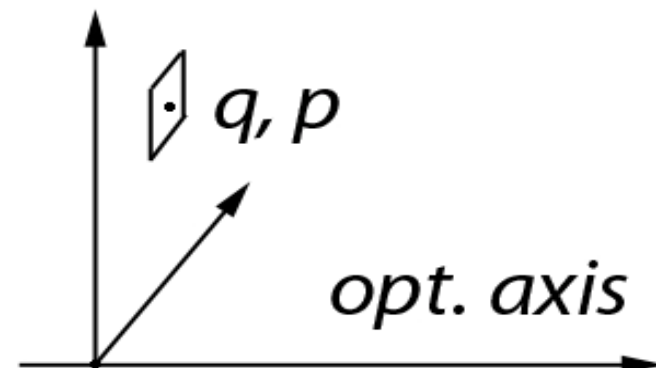
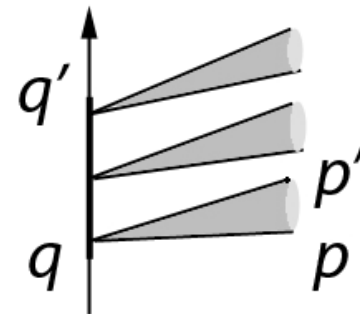
Radiance

Definition and main mathematical properties



Radiance (Definition)

- ▶ Energy density in 4D ray-space
- ▶ Energy in all small p -cones (each one from p to p') in a small q -box (from q to q'), perpendicular to p
- ▶ Energy in a small 4D cube around (q, p) , divided by the volume
- ▶ Intuitively it is the “light-field” or “light itself”
- ▶ Express radiance as $r(q,p)$

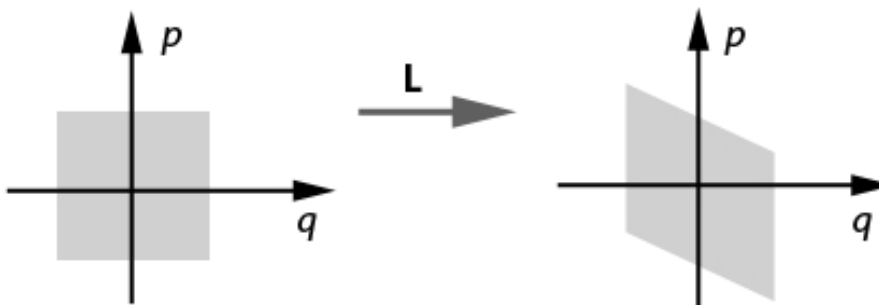




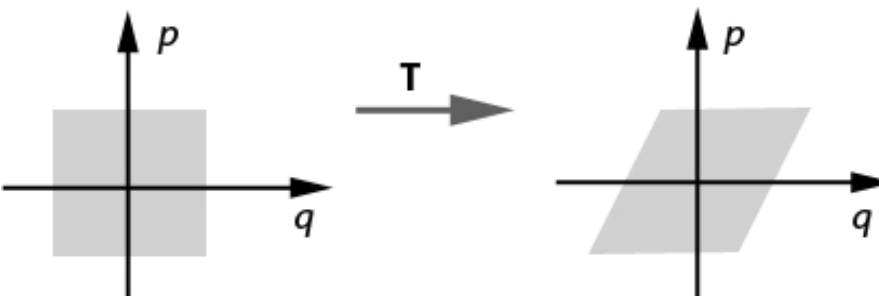
Conservation of Volume

- ▶ For the 2 transforms, the 4D box changes shape
- ▶ Volume remains the same (shear)
- ▶ Must remain the same for any optical transform!

$$L = \begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix}$$



$$T = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix}$$





Conservation of Radiance

- ▶ Radiance is energy density in 4D ray-space
- ▶ Energy is conserved; volume is conserved
- ▶ Radiance = (energy) / (volume)
- ▶ ***Radiance is also conserved!***
- ▶ ***“Radiance is constant along each ray”***



Additional Notes on Conservation of Radiance

- ▶ Similar principle in Hamiltonian mechanics in terms of coordinate q and momentum p : Liouville's theorem
- ▶ As the system evolves in time, volume in qp -space is conserved
 - ▶ State space and particle systems
 - ▶ Quantum mechanics
- ▶ In optics, astronomy, and photography, radiance conservation is often mentioned (or implied) in relation to:
 - ▶ Throughput
 - ▶ Barlow lens
 - ▶ Teleconverter
 - ▶ F/number



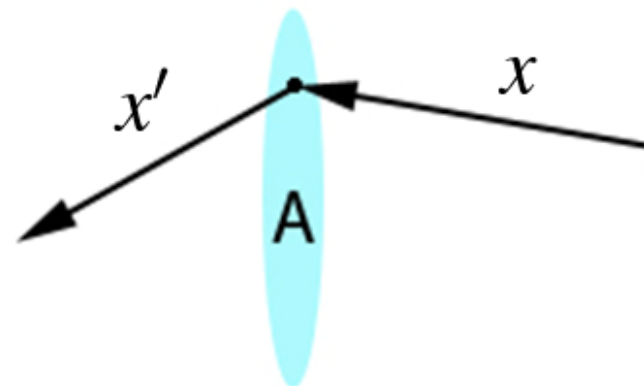
Additional Notes on Conservation of Radiance

- ▶ Optical state space is a vector space with the optical axis being the zero vector
 - ▶ Optical devices, like cameras and microscopes perform linear transforms.
- ▶ Optical transforms are symplectic:
 - ▶ They preserve a skew-symmetric dot product in qp -space
 - ▶ In terms of that dot product each ray is orthogonal to itself
- ▶ For any optical transform A , $\det A = 1$



Radiance Transforms

- ▶ Optical elements transform rays
- ▶ They also transform radiance



- ▶ Points in ray space $x = \begin{bmatrix} q \\ p \end{bmatrix}$
- ▶ Radiance before optical transform $r(x)$
- ▶ Radiance after optical transform $r'(x)$



Radiance Transforms

$$x' = Ax$$

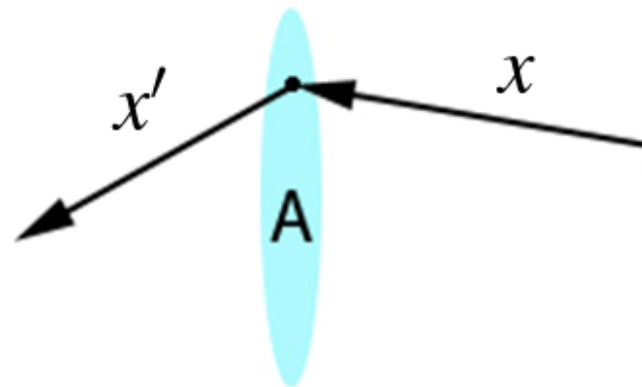
Due to radiance conservation,

$$r'(x') = r(x)$$

$$r'(x') = r(A^{-1}x')$$

Since x' is arbitrary, we can replace it by x

$$r'(x) = r(A^{-1}x)$$





Radiance Transforms

- ▶ The radiance after optical transformation is related to the original radiance by: $r'(x) = r(A^{-1}x)$

- ▶ What is that for translation?

$$T = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix} \quad T^{-1} = \begin{bmatrix} 1 & -t \\ 0 & 1 \end{bmatrix}$$

- ▶ So $r'(q, p) = r(q - tp, p)$



In-Class Exercise

- ▶ The radiance after optical transformation is related to the original radiance by: $r'(x) = r(A^{-1}x)$

- ▶ What is that for a lens?

$$L = \begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix} \quad L^{-1} = \begin{bmatrix} 1 & 0 \\ \frac{1}{f} & 1 \end{bmatrix}$$

- ▶ So $r'(q, p) = r(q, p + \frac{q}{f})$

Image Rendering

- ▶ Now that we have the lightfield (all of the light in a scene) – how do we turn q and p into a picture?
 - ▶ (A rendered image)?
- ▶ Use physics of
- ▶ integral image
- ▶ formation

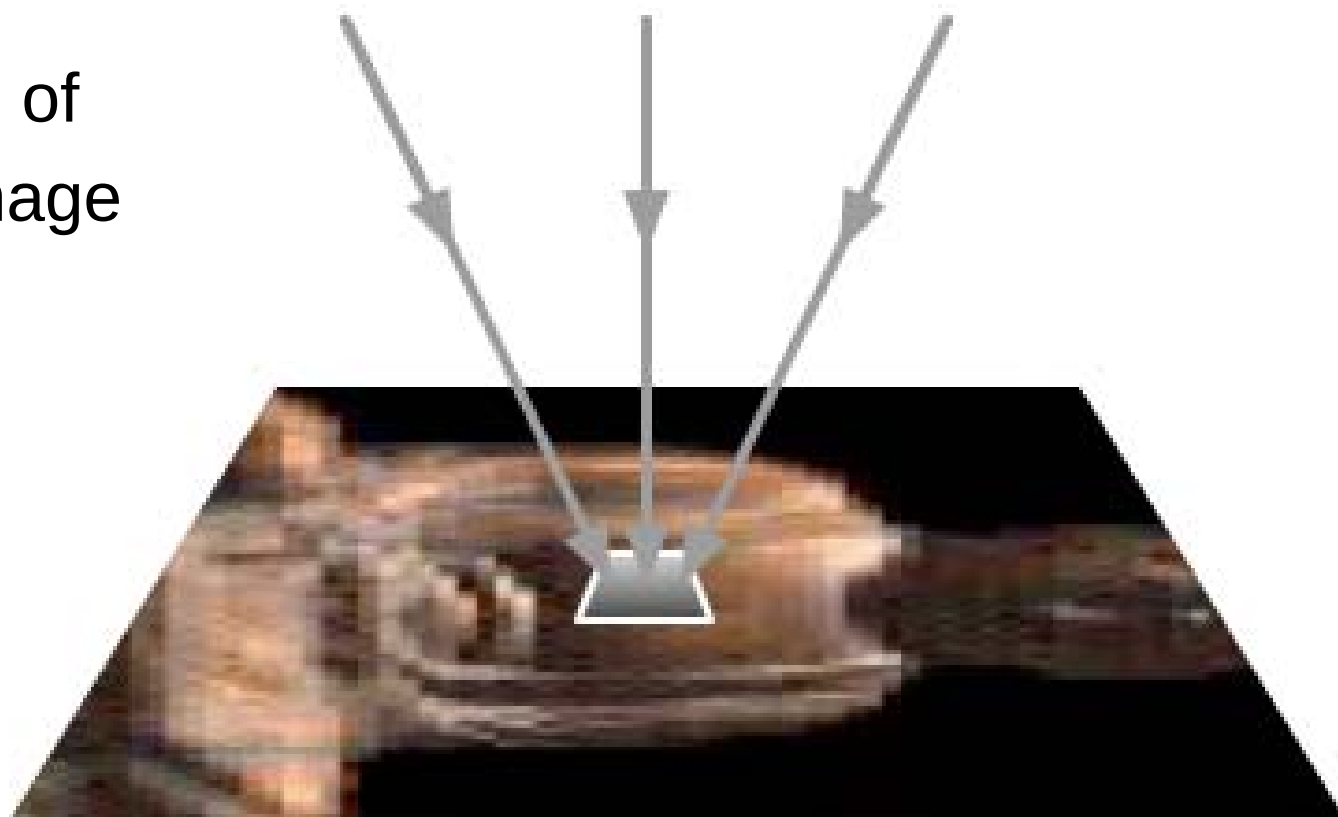
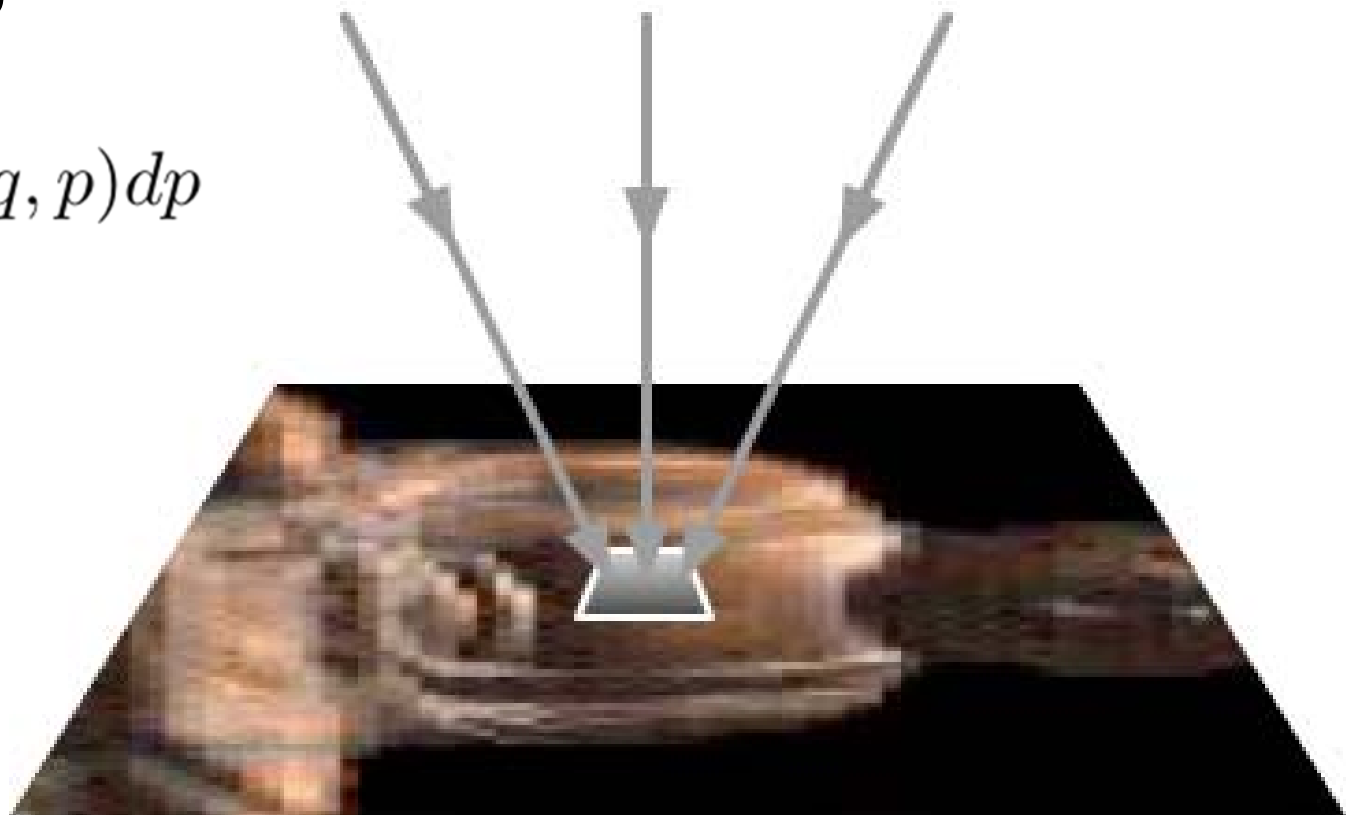


Image Rendering

- ▶ A traditional image is formed by integrating rays from all directions at each pixel
- ▶ A traditional image is rendered from a radiance according to

$$I(q) = \int_p r(q, p) dp$$

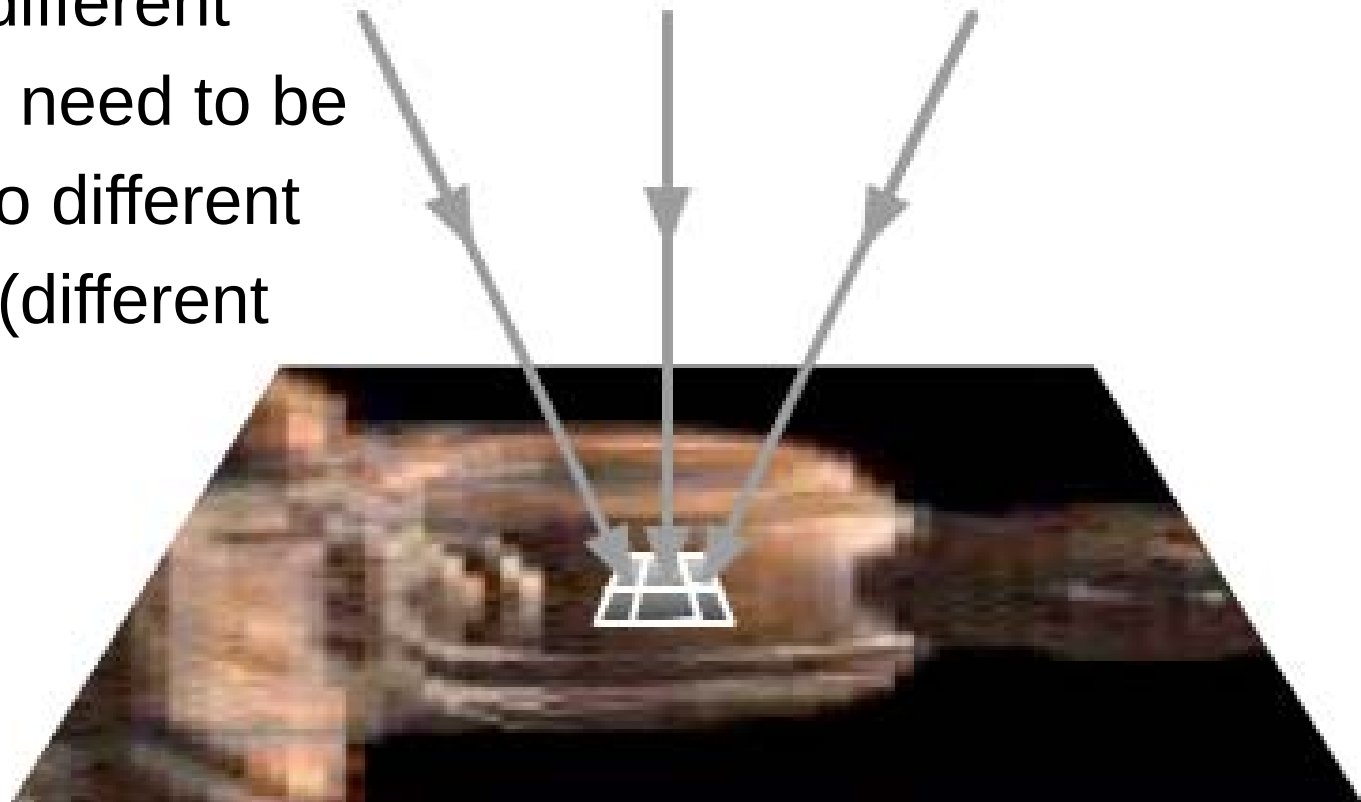


Capturing Radiance with Cameras



Capturing Radiance

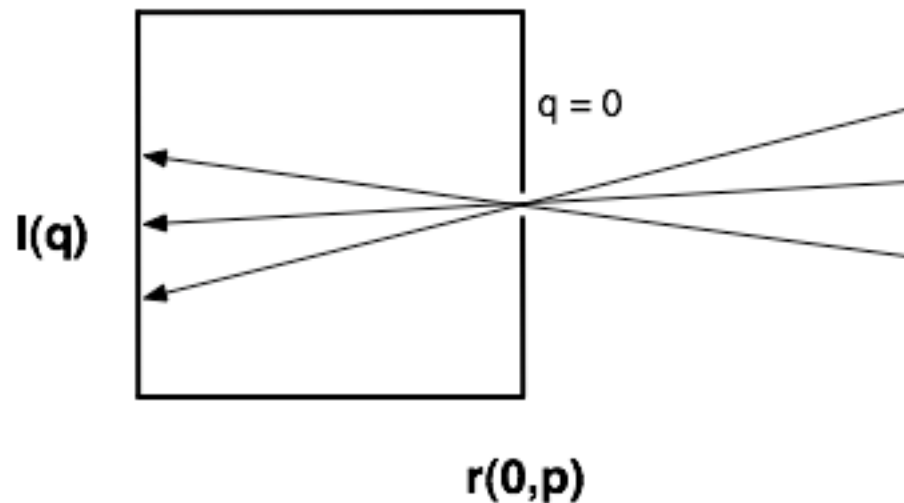
- ▶ To capture radiance, we need to capture rays from different directions individually
- ▶ But sensors are not directional
- ▶ Rays from different
- ▶ directions need to be
- ▶ mapped to different
- ▶ positions (different
- ▶ pixels)





Pinhole Camera

- ▶ Rays can only enter camera at one point ($q = 0$)
- ▶ Rays from different directions spread apart inside camera
- ▶ And are captured at different positions on the sensor

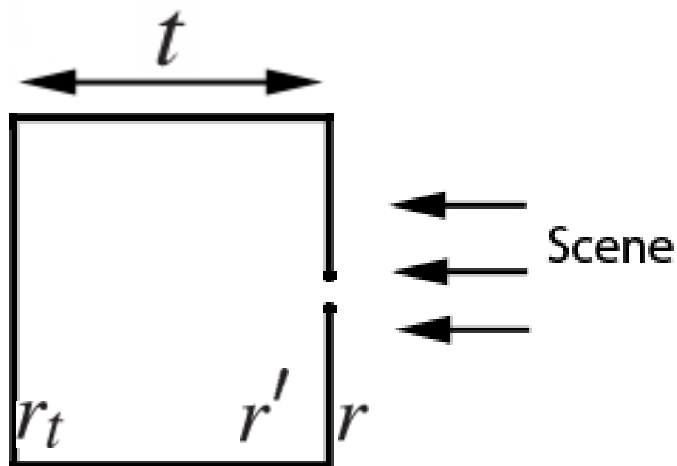


- ▶ Switches direction and position
- ▶ Captures angular distribution of radiance



Pinhole Camera

- ▶ More precisely



$$r'(q, p) = r(q, p)\delta(q)$$

$$r_t(q, p) = r(q - tp)\delta(q - tp)$$

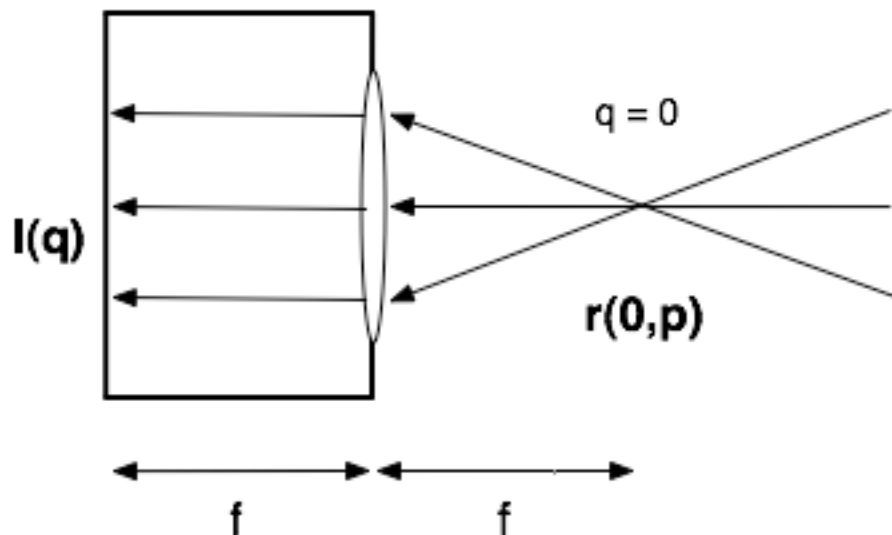
$$I(q) = \int r_t(q, p)dp = \frac{1}{t} \int r(q - tp, \frac{tp}{t})\delta(q - tp)d(tp) = \frac{1}{t}r(0, \frac{q}{t})$$

- ▶ Switches angle and position
- ▶ Captures angular distribution of radiance



“2F” Camera

- ▶ Generalizes pinhole camera
- ▶ Lens of focal length f is placed at distance f from sensor

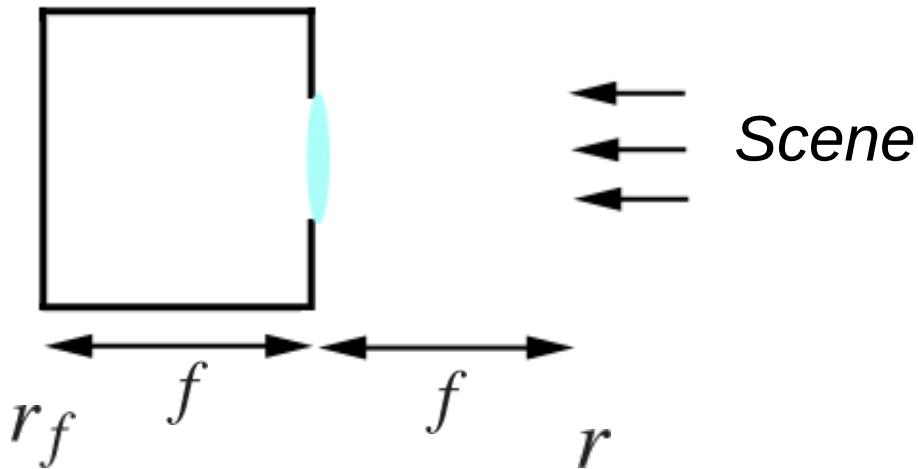


- ▶ Switches angle and position
- ▶ Captures angular distribution of radiance assuming it doesn't change much with q (close to $q = 0$)



“2F” Camera

- ▶ This is the lens generalization of the pinhole camera
- ▶ Three optical elements: space, lens, space

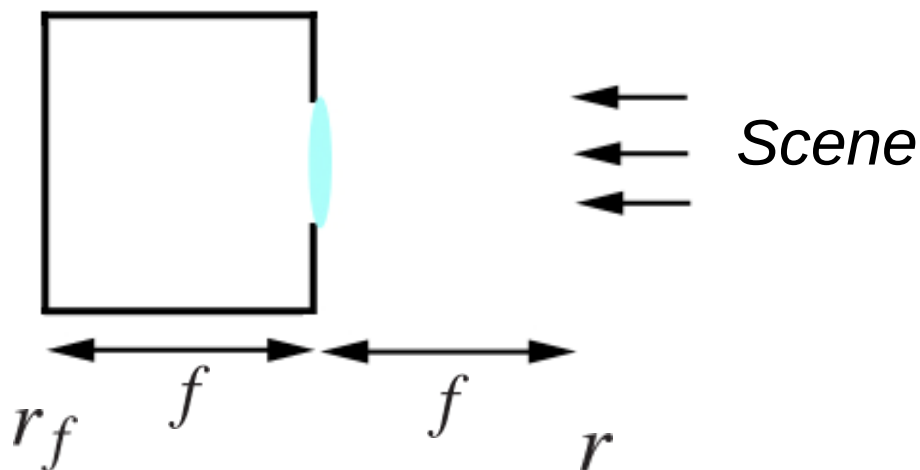


- ▶ Transformation: $A = T_f L_f T_f$



“2F” Camera

- ▶ This is the lens generalization of the pinhole camera
- ▶ Three optical elements: space, lens, space



$$A = \begin{bmatrix} 0 & f \\ -\frac{1}{f} & 0 \end{bmatrix}$$

Show that $I(q) = \frac{D}{f} r(0, \frac{q}{f})$

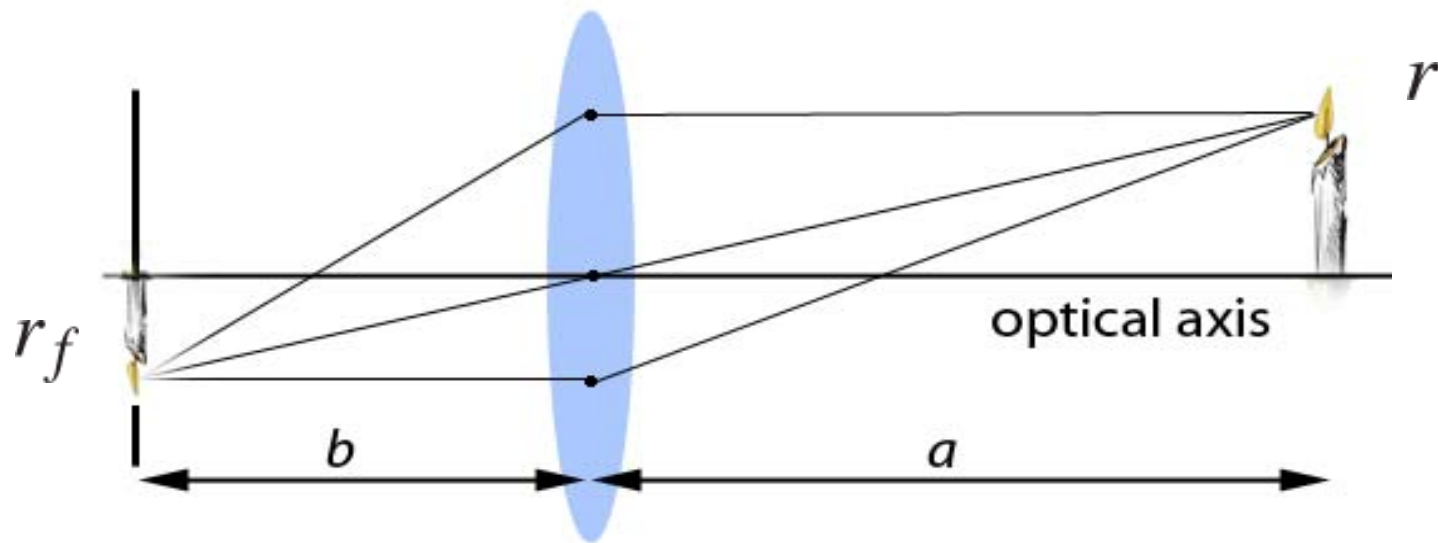
D is the aperture diameter

- ▶ Switches angle and position
- ▶ Captures angular distribution of radiance (at $q = 0$)



Traditional 2D Camera

Three optical elements: **space, lens, space**



$$A = \begin{bmatrix} -\frac{b}{a} & 0 \\ -\frac{1}{f} & -\frac{a}{b} \end{bmatrix}$$

Show that $I(q) = \frac{D}{b} r(-\frac{a}{b}q, 0)$ approximately.

Capturing Radiance

- ▶ Pinhole camera or “2F” camera capture an image $I(q)$
- ▶ $I(q)$ captures angular distribution of radiance

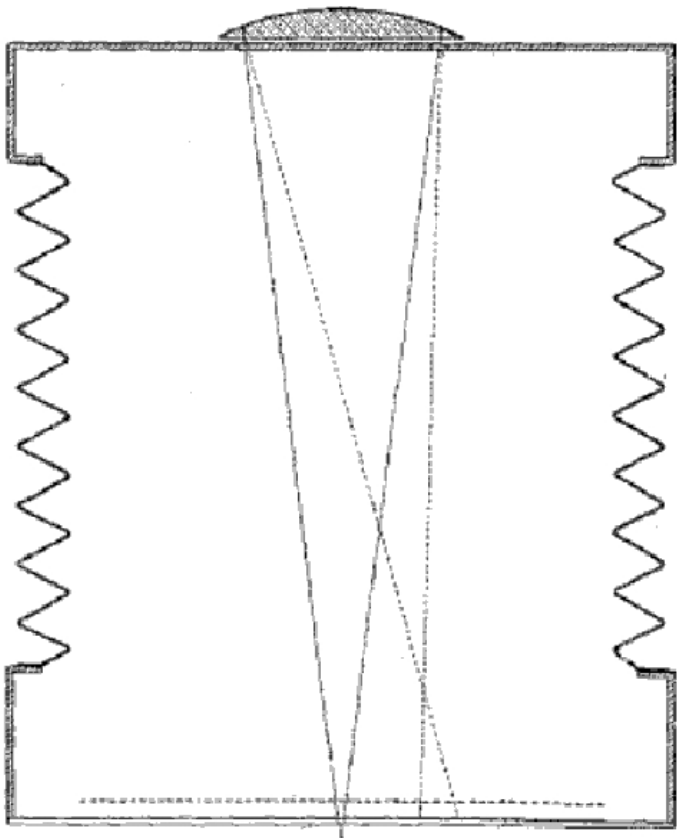
$$I(q) = \frac{1}{t} r(0, \frac{q}{t})$$

$$I(q) = \frac{D}{f} r(0, \frac{q}{f})$$

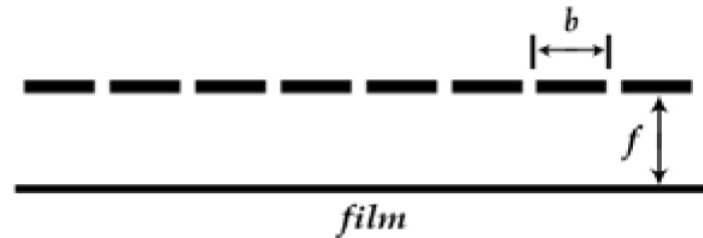
- ▶ Only for small area around $q = 0$ so far
- ▶ For complete radiance, we need to capture angular distribution for all q
- ▶ Basic Idea: ***Replicate pinhole or “2F” at every q***
 - ▶ Ives (pinhole)
 - ▶ Lippmann (“2F”)



Ives' Camera (based on the pinhole camera)



At the image plane:



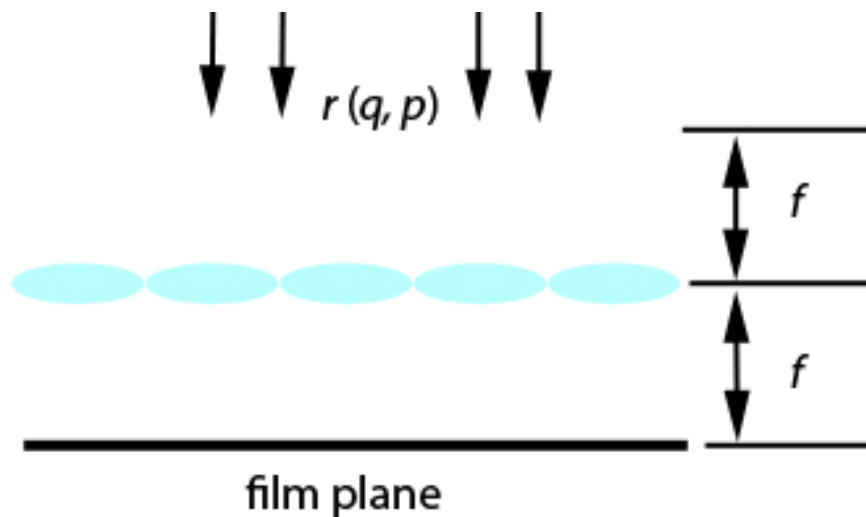
Multiplexing in space:

Each pinhole image captures angular distribution of radiance. All images together describe the complete 4D radiance.



Lippmann's Camera (based on 2F)

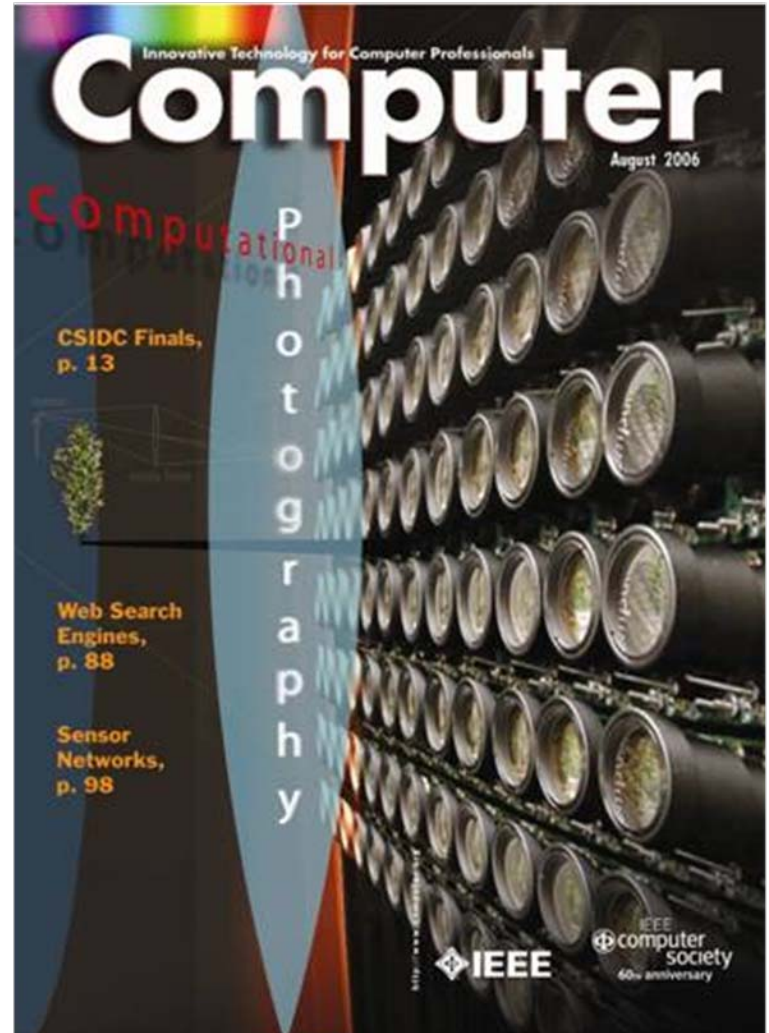
- ▶ Space multiplexing
- ▶ Lenses instead of pinholes
- ▶ A “2F camera” replaces each pinhole camera in Ives’ design





Camera Arrays

- ▶ The most popular lightfield camera is simply an array of conventional cameras, like the Stanford array.
- ▶ Alternatively, an array of lenses/prisms with a common sensor, like the Adobe array.





Adobe Array of Lenses and Prisms





Arrays of Lenses and Prisms

Shifting cameras from the optical axis means: We need to extend the vector space treatment to affine space treatment.

Prism transform

$$\begin{pmatrix} q' \\ p' \end{pmatrix} = \begin{pmatrix} q \\ p \end{pmatrix} + \begin{pmatrix} 0 \\ \alpha \end{pmatrix}$$

Shifted lens

$$\begin{pmatrix} q' \\ p' \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{pmatrix} \begin{pmatrix} q-s \\ p \end{pmatrix} + \begin{pmatrix} s \\ 0 \end{pmatrix}$$

Lens + prism

$$\begin{pmatrix} q' \\ p' \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{pmatrix} \begin{pmatrix} q \\ p \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{s}{f} \end{pmatrix}$$



Refocusing with the Adobe camera





Refocusing with the Adobe camera





Radiance in the Frequency Domain

In the frequency domain, the two optical elements switch places:
lens becomes space; space becomes lens

Radiance Transforms (Frequency Domain)

- ▶ Converting radiance into frequency representation gives us a new tool for analysis, and new power.
- ▶ A pixel no longer stands by itself, representing a point in one single image / slice in 4D radiance.
- ▶ In the frequency domain one pixel can represent multiple images at the same time.
- ▶ Those images are slices of the 4D radiance, but now in the frequency domain.
- ▶ By optically combining multiple frequencies, we achieve new and more efficient use of the sensor.

Radiance Transforms (Frequency Domain)

Radiance in frequency representation:

$$R(\omega) = \int r(x) e^{i\omega \cdot x} dx$$

where $\omega = \begin{bmatrix} \omega_q \\ \omega_p \end{bmatrix}$ and $\omega \cdot x = \omega_q q + \omega_p p$

Next we derive the relation between $R'(\omega)$
and $R(\omega)$ due to optical transform $x = Ax_0$



Radiance Transforms (Frequency Domain)

$$\begin{aligned} R'(\omega) &= \int r'(x) e^{i\omega \cdot x} dx \\ &= \int r(A^{-1}x) e^{i\omega \cdot x} dx \\ &= \int r(A^{-1}x) e^{i\omega \cdot AA^{-1}x} dx \\ &= \int r(x_0) e^{i\omega \cdot Ax_0} dx_0 \\ &= \int r(x_0) e^{iA^t \omega \cdot x_0} dx_0 \\ &= R(A^T \omega) \end{aligned}$$



Radiance Transforms

Main results (summary):

$$\begin{aligned}x &= Ax_0 \\ r'(x) &= r(A^{-1}x) \\ R'(\omega) &= r(A^T\omega)\end{aligned}$$

Note: Shear is in the other direction in frequency domain due to the transposed matrix. Lens $\leftrightarrow$ space.

Note: The inverse always exists because $\det A = 1$.



Ives' Camera: Frequency Multiplexing

- Poisson summation formula

$$\sum_{m=-\infty}^{\infty} \delta(q - m) = \sum_{n=-\infty}^{\infty} e^{in2\pi q}$$

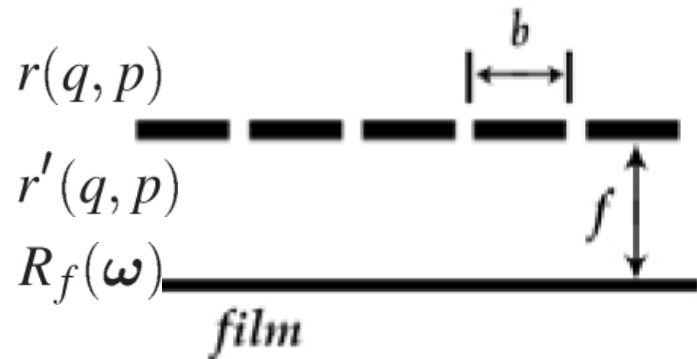
“train of delta functions = train of frequencies”

Prove

$$\sum_{m=-\infty}^{\infty} \delta(q - mb) = \frac{1}{b} \sum_{n=-\infty}^{\infty} e^{in2\pi \frac{q}{b}}$$



Ives' Camera: Frequency Multiplexing


$$r'(q, p) = r(q, p) \sum_{m=-\infty}^{\infty} \delta(q - mb)$$

$$\begin{aligned} R'(\omega) &= \int r(q, p) \sum_m \delta(q - mb) e^{i\omega \cdot x} dx \\ &= \frac{1}{b} \int r(q, p) \sum_n e^{in \frac{2\pi q}{b}} e^{i(\omega_q q + \omega_p p)} dq dp \\ &= \frac{1}{b} \sum_n R(\omega_q + n \frac{2\pi}{b}, \omega_p). \end{aligned}$$

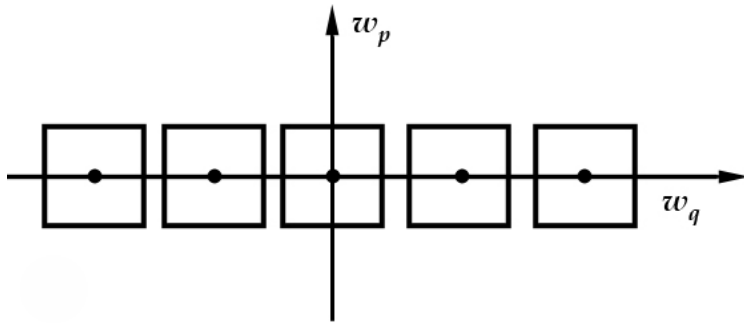
Transposed translation f

$$\begin{pmatrix} 1 & 0 \\ f & 1 \end{pmatrix} \begin{pmatrix} \omega_q \\ \omega_p \end{pmatrix} = \begin{pmatrix} \omega_q \\ f\omega_q + \omega_p \end{pmatrix}$$

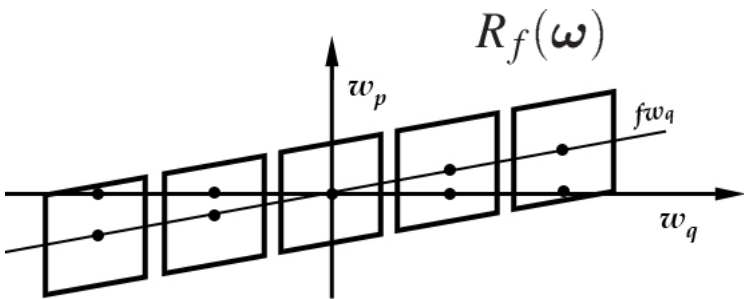
$$R_f(\omega) = \sum_{n=-\infty}^{\infty} R(\omega_q + n \frac{2\pi}{b}, f\omega_q + \omega_p)$$



Ives' Camera: Frequency Multiplexing

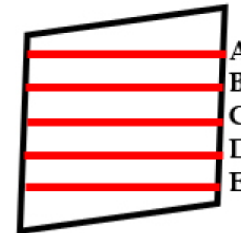
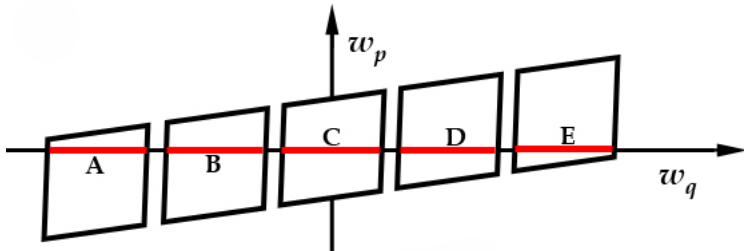


Band limited radiance



$$R_f(\omega) = \sum_{n=-\infty}^{\infty} R(\omega_q + n \frac{2\pi}{b}, f\omega_q + \omega_p)$$

Veeraraghavan's idea:





Cosine Mask Camera (MERL)

- ▶ A transparency, superposition of $\cos$ terms, is placed at distance f from the sensor
- ▶ Consider for example:

$$\frac{1}{2}(1 + \cos(\omega_0 q))$$

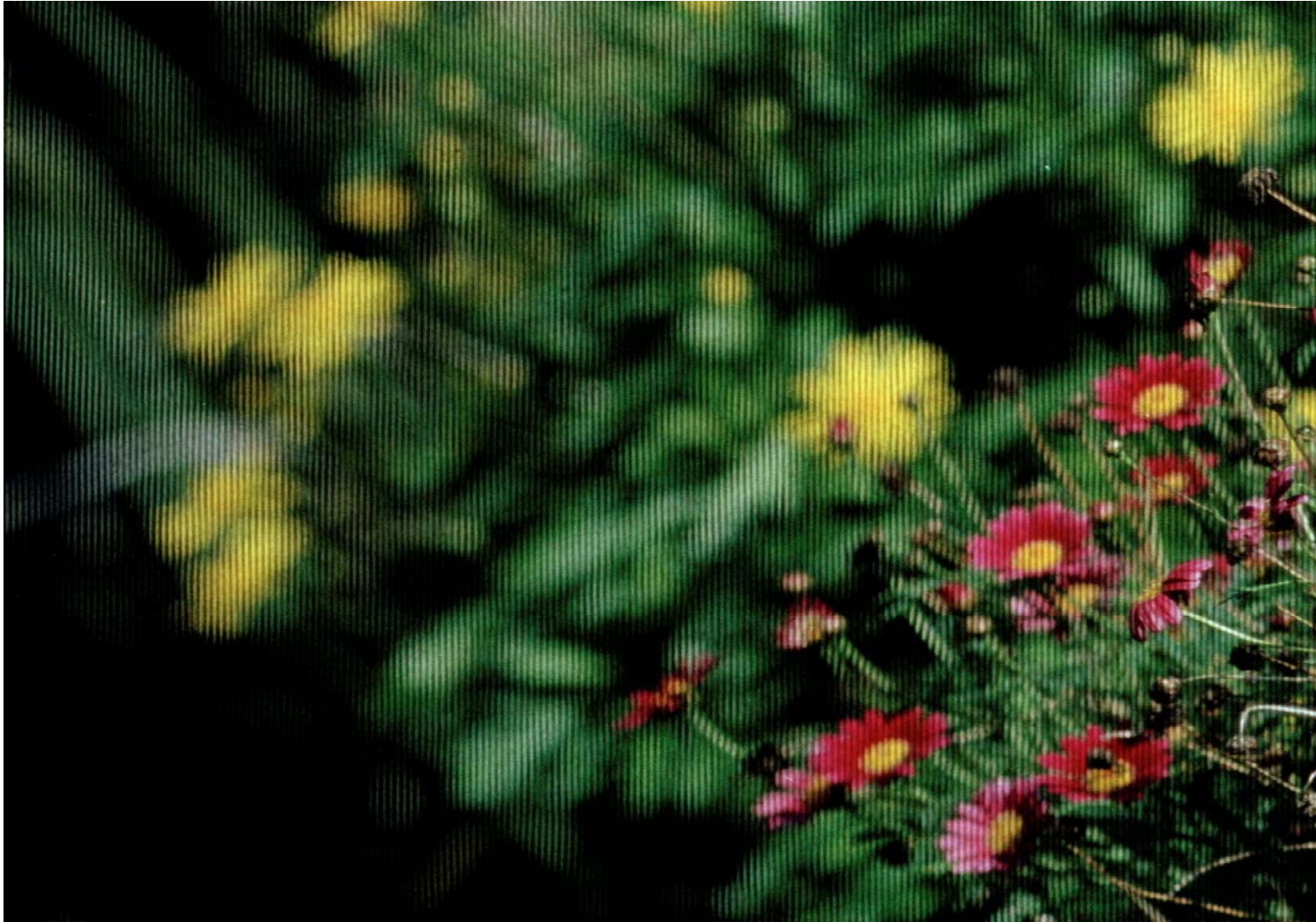
Derive the expression for the radiance at the sensor:

$$\begin{aligned} R_f(\omega_q, \omega_p) &= \frac{1}{2}R(\omega_q, f\omega_q + \omega_p) \\ &+ \frac{1}{4}(R(\omega_q + \omega_0, f\omega_q + \omega_p) + R(\omega_q - \omega_0, f\omega_q + \omega_p)) \end{aligned}$$



Periodic Mask Camera (Adobe)

Input:
F/5.6





Periodic Mask Camera (Adobe)

**Output:
F/5.6**





Periodic Mask Camera (Adobe)

**Output:
F/5.6**



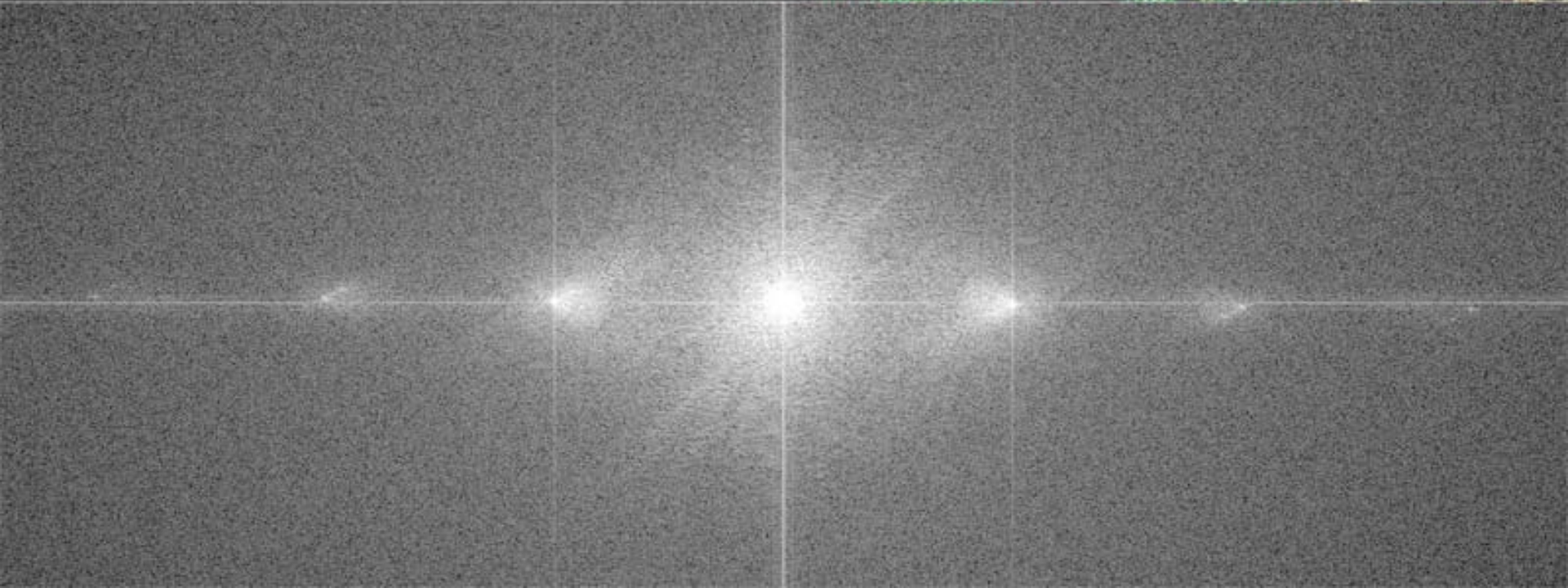


F/11



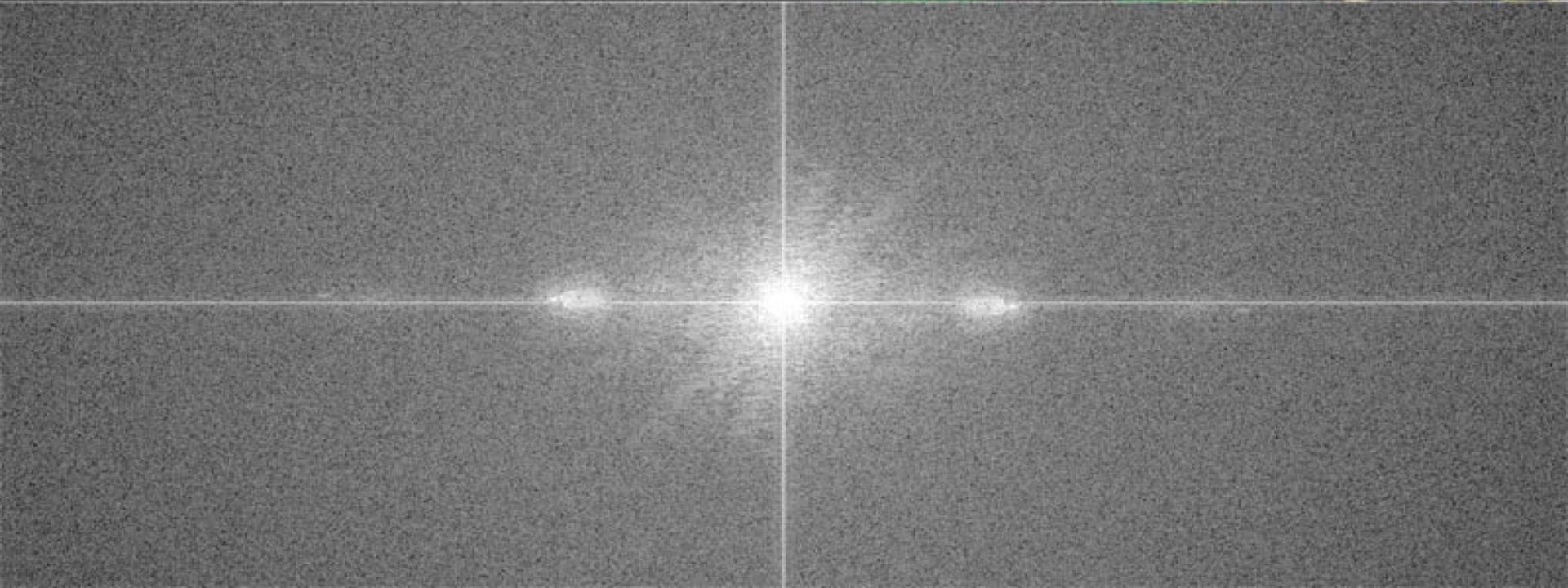


F/5.6





F/4





Periodic Mask Camera (Adobe)

**Output:
F/4**





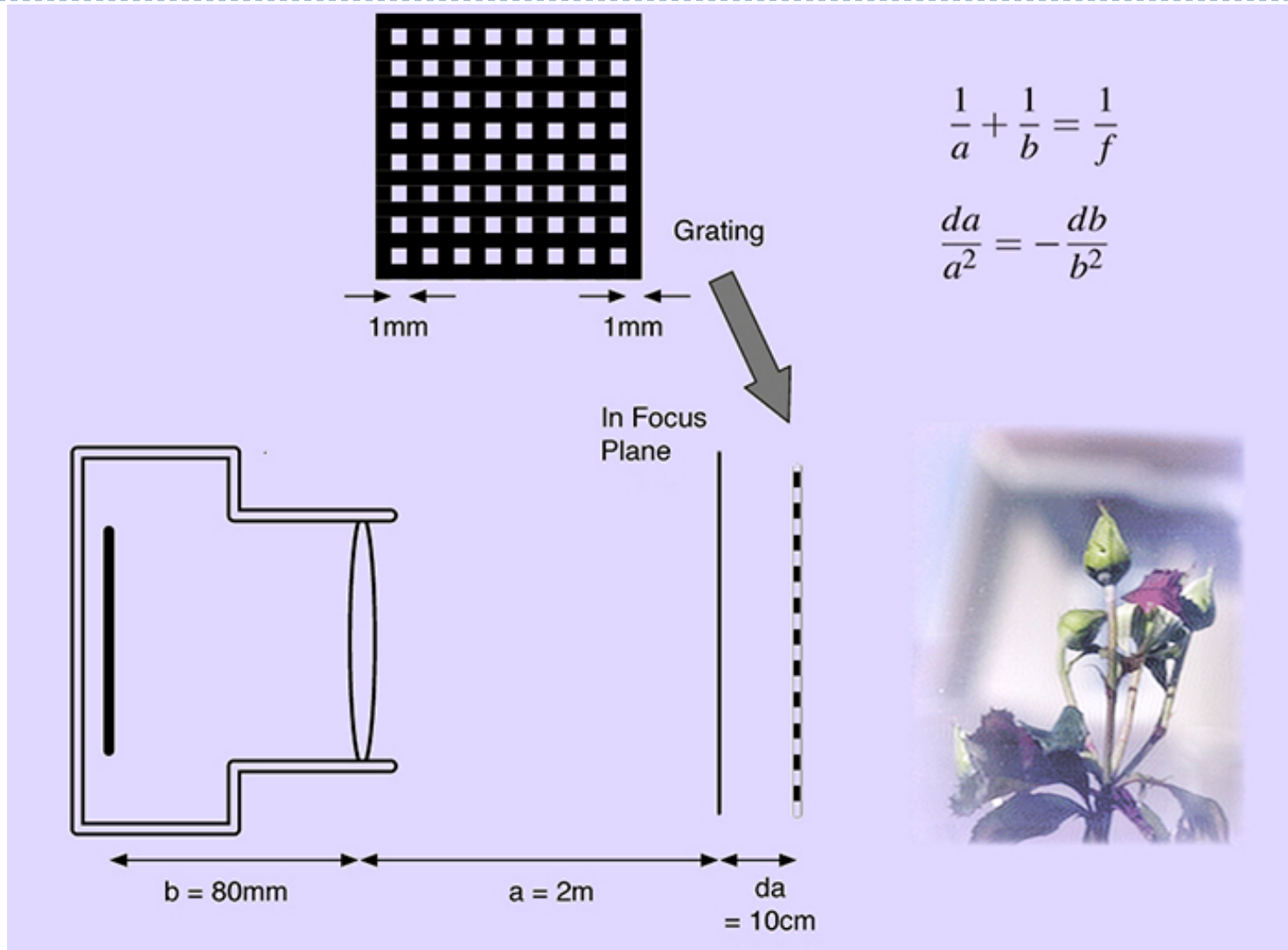
Periodic Mask Camera (Adobe)

**Output:
F/4**





“Mosquito Net” Camera



“Mosquito Net Camera” Refocusing



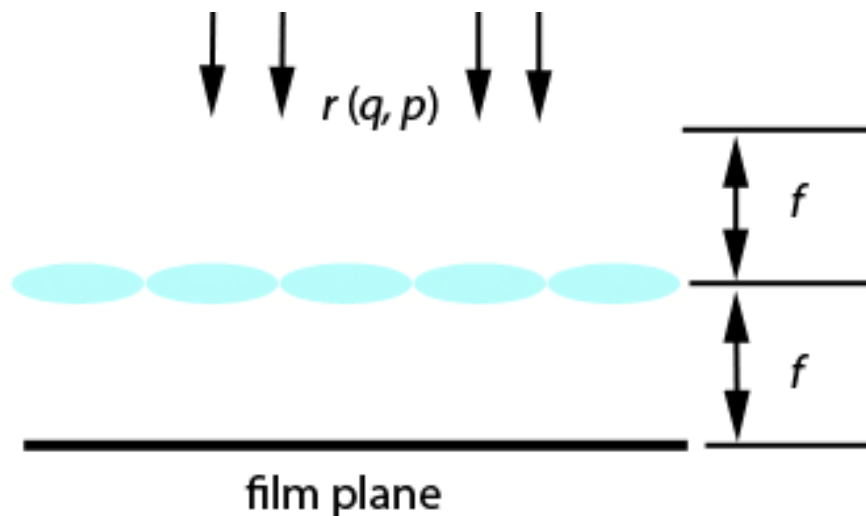
“Mosquito Net Camera” Refocusing





Lippmann's Camera

- ▶ Space multiplexing
- ▶ Lenses instead of pinholes
- ▶ A “2F camera” replaces each pinhole camera in Ives’ design

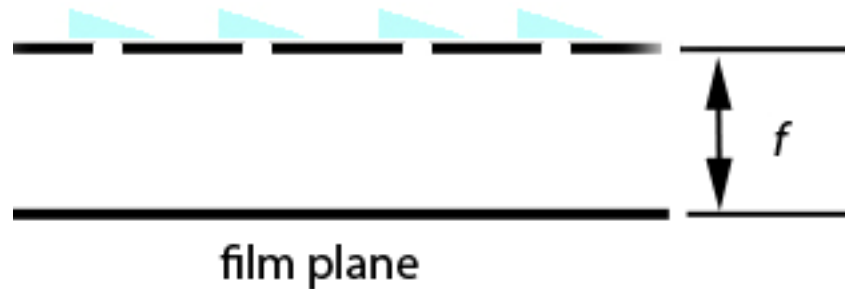




Lippmann's Camera – “Heterodyning”

- ▶ Frequency multiplexing or “heterodyning” analysis can be done in two steps:

1. Consider array of shifted pinhole-prisms with constant shift a , and prism angle a/f



2. Superposition of arrays with different shifts to implement microlenses as Fresnel lenses.



Lippmann's Camera – “Heterodyning”

- ▶ Starting with

$$R(\omega) = \int r(q, p + \frac{a}{f}) \sum_m \delta(q - mb - a) e^{i\omega \cdot x} dx$$

- ▶ Derive the radiance at the focal plane
- ▶ Show that at zero angular frequency it becomes:

$$R_f(\omega_q, 0) = \frac{1}{b} \sum_n e^{-i(\omega_q a + n \frac{2\pi a}{b})} R(\omega_q + n \frac{2\pi}{b}, f\omega_q)$$



Lippmann's Camera – “Heterodyning”

$$\begin{aligned} R'(\omega) &= \int r(q, p) \sum_m \delta(q - mb) e^{i\omega \cdot \mathbf{x}} d\mathbf{x} \\ &= \frac{1}{b} \int r(q, p) \sum_n e^{in\frac{2\pi q}{b}} e^{i(\omega_q q + \omega_p p)} dq dp \\ &= \frac{1}{b} \sum_n R(\omega_q + n\frac{2\pi}{b}, \omega_p). \end{aligned}$$

$$R_f(\omega) = \sum_{n=-\infty}^{\infty} R(\omega_q + n\frac{2\pi}{b}, f\omega_q + \omega_p)$$



Lippmann's Camera – “Heterodyning”

$$\begin{aligned} R'(\omega) &= \int r(q, p + \frac{a}{f}) \sum_m \delta(q - mb - a) e^{i\omega \cdot x} dx \\ &= \frac{1}{b} \int r(q, p + \frac{a}{f}) \sum_n e^{in \frac{2\pi(q-a)}{b}} e^{i(\omega_q q + \omega_p p)} dq dp \\ &= \frac{1}{b} \sum_n e^{-i(\omega_p \frac{a}{f} + n \frac{2\pi a}{b})} R(\omega_q + n \frac{2\pi}{b}, \omega_p). \end{aligned}$$

$$R_f(\omega) = \frac{1}{b} \sum_n e^{-i((f\omega_q + \omega_p) \frac{a}{f} + n \frac{2\pi a}{b})} R(\omega_q + n \frac{2\pi}{b}, f\omega_q + \omega_p)$$

$$R_f(\omega_q, 0) = \frac{1}{b} \sum_n e^{-i(\omega_q a + n \frac{2\pi a}{b})} R(\omega_q + n \frac{2\pi}{b}, f\omega_q)$$



Lippmann's Camera – “Heterodyning”

Plenoptic (Integral) camera with frequency multiplexing



Thanks to Ren Ng for providing the lightfield image.



The Focused Plenoptic Camera

“Lightfield photographers, focus your cameras!”

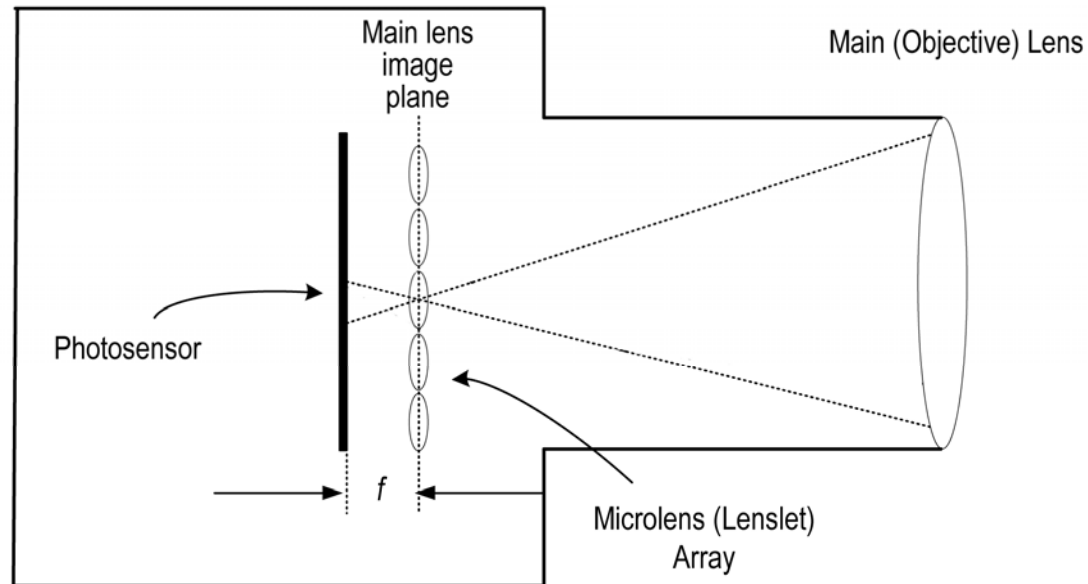
Karl Marx



Plenoptic Camera, Adelson 1992

- ▶ Main lens focused on microlenses

Conventional plenoptic camera

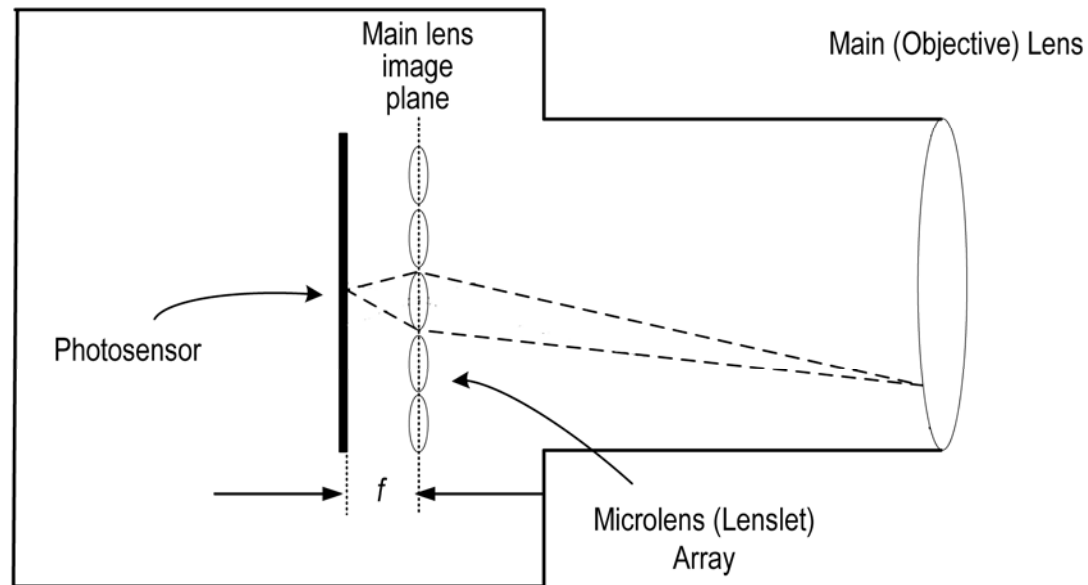




Plenoptic Camera, Adelson 1992

- Microlenses focused on infinity

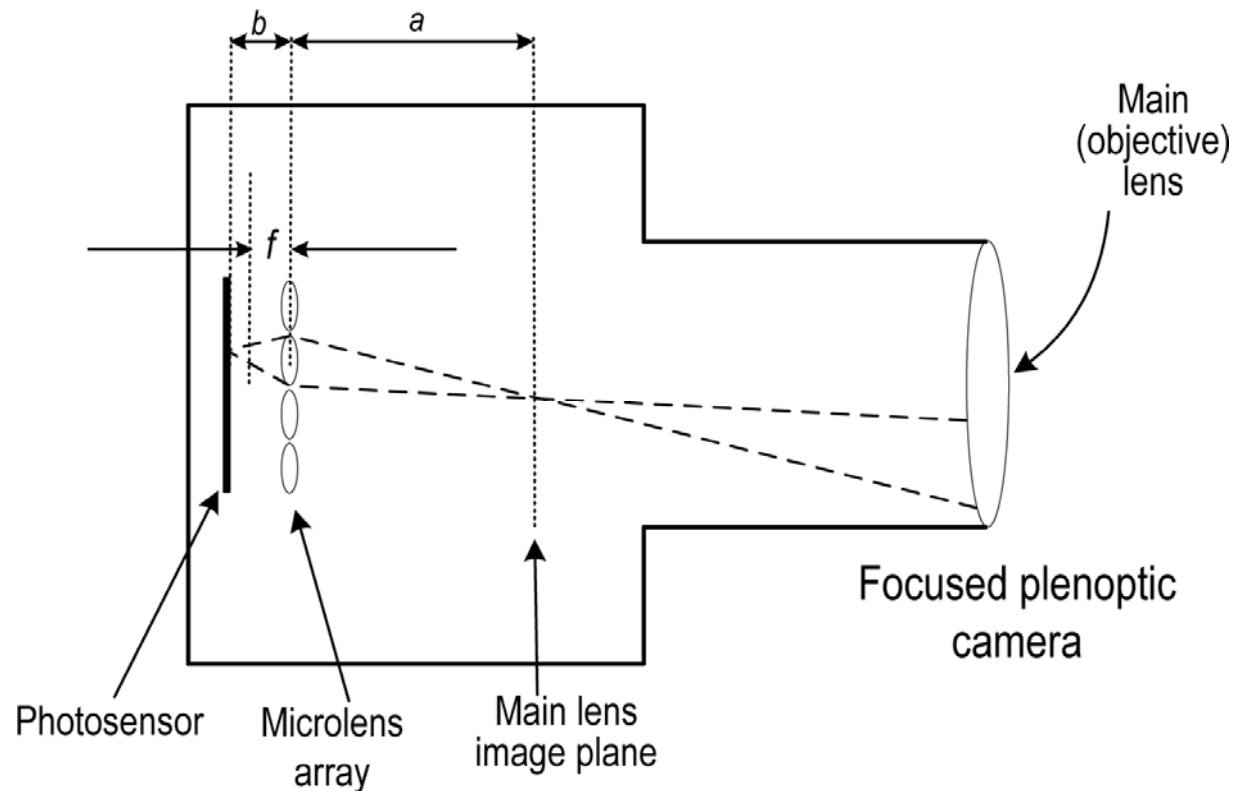
Conventional plenoptic camera





Focused Plenoptic Camera

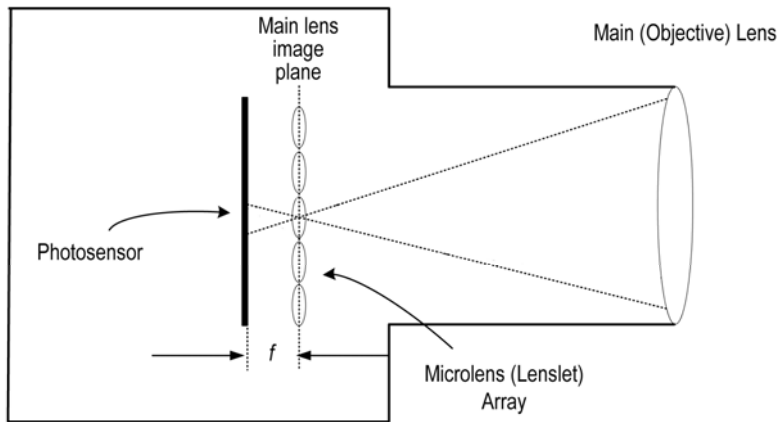
- ▶ Microlenses focused on main lens image



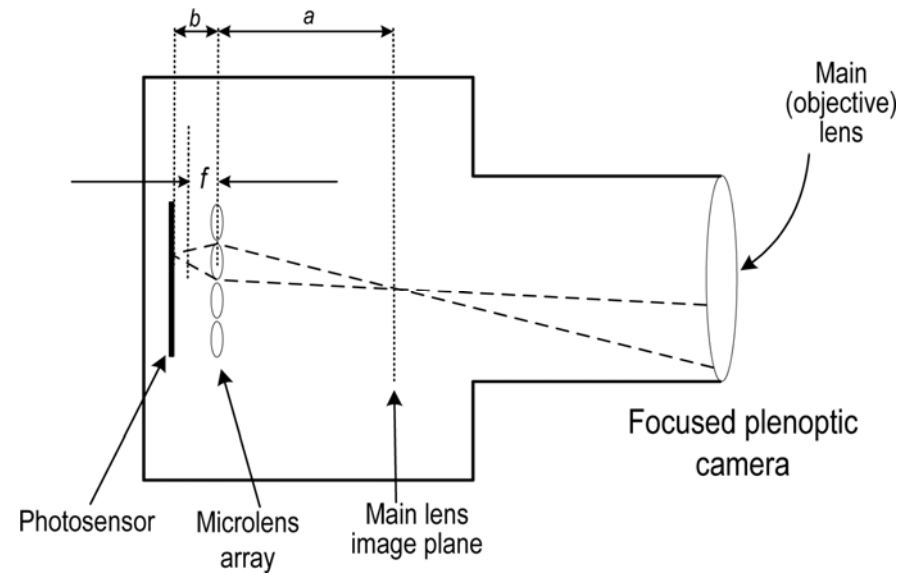
Comparison

► Plenoptic Camera (1.0)

Conventional plenoptic camera

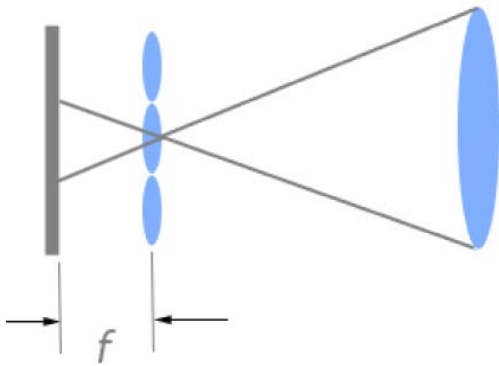


► Focused Plenoptic Camera (2.0)



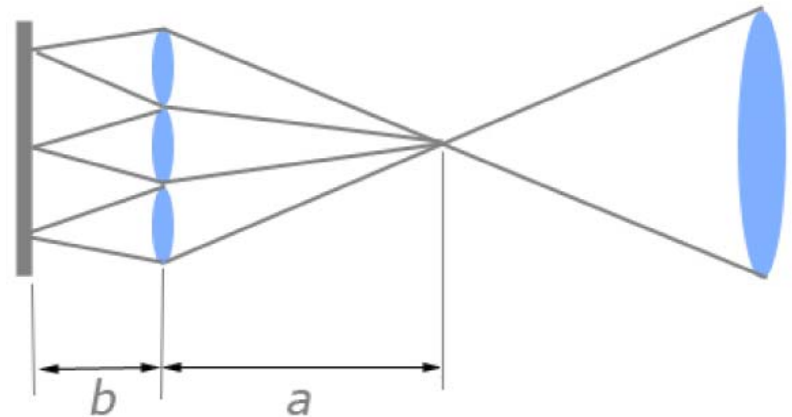
Comparison

► Plenoptic Camera



Microlenses focused at infinity. Completely defocused relative to main lens image.

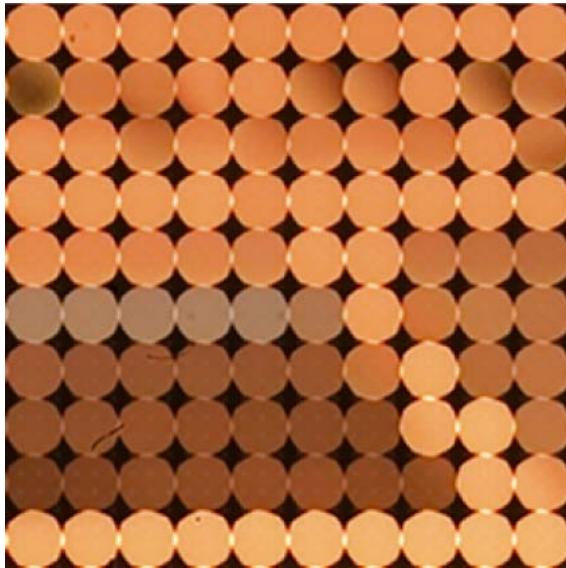
► Focused Plenoptic Camera



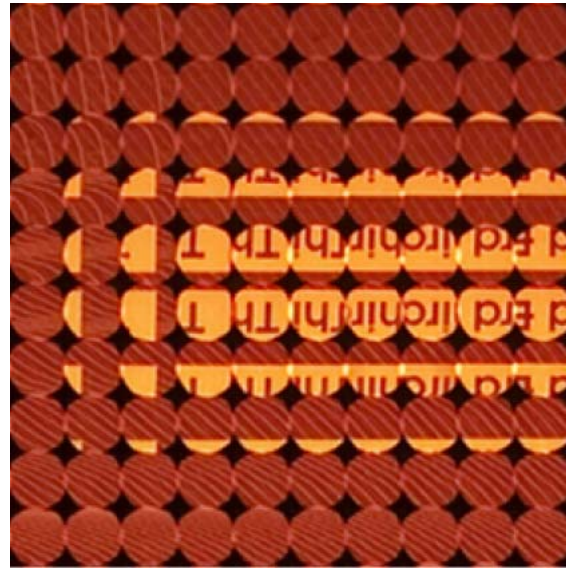
Microlenses satisfy the lens equation. Exactly focused on the main lens image.

Comparison

- ▶ Plenoptic Camera
Blurry microimages



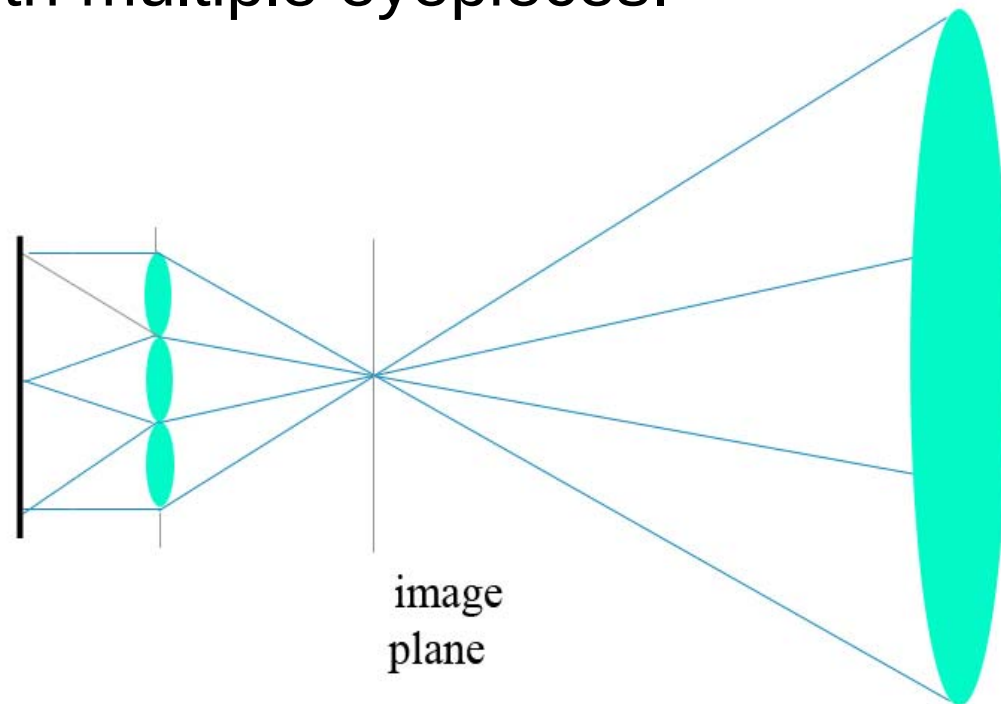
- ▶ Focused Plenoptic Camera
Sharp and inverted microimages





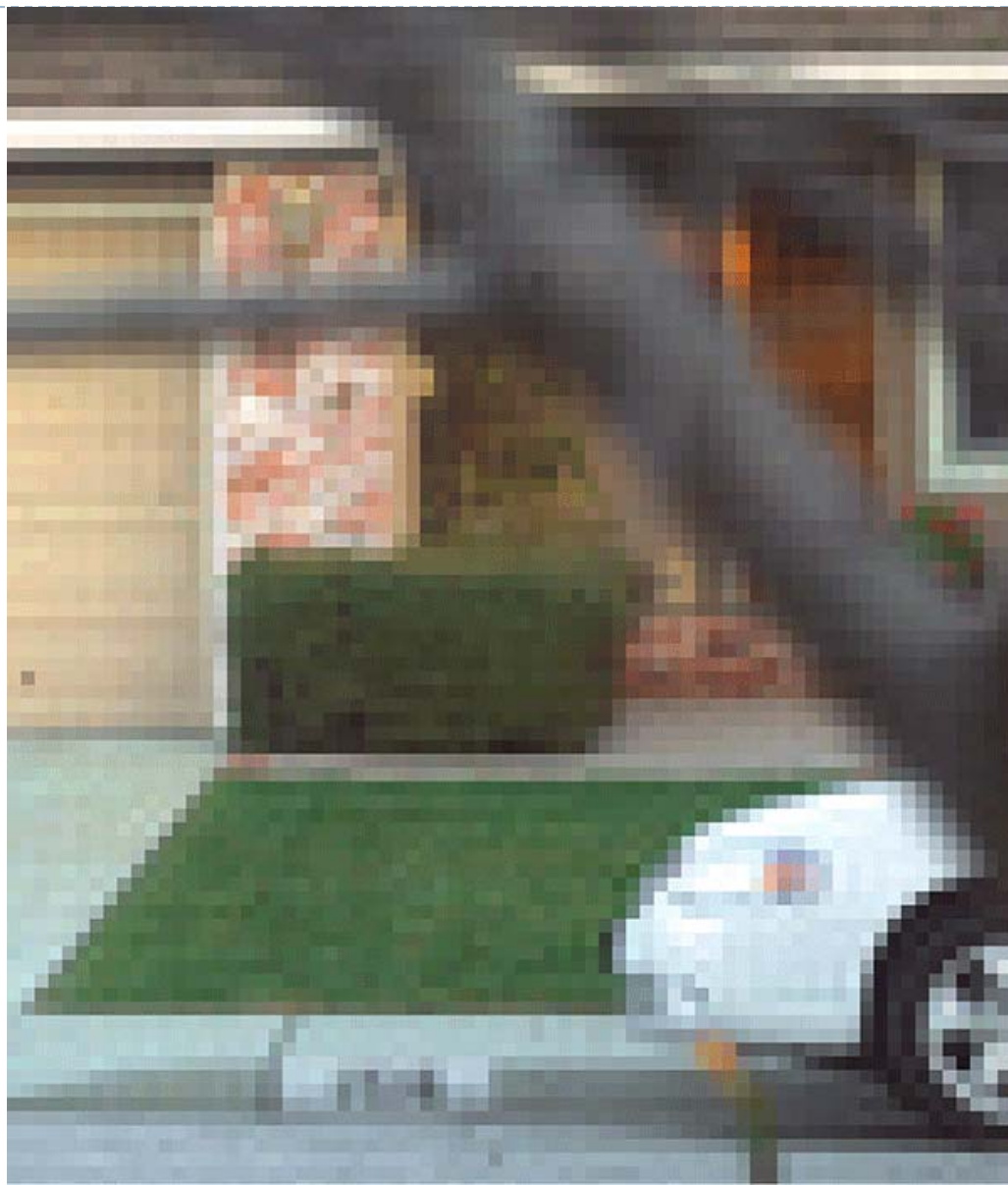
Why Inverted?

- ▶ What is the condition for exact focusing with main lens image shifted from the plane of microlenses?
- ▶ **Answer:** Simple relay imaging! This is like a telescope with multiple eyepieces.





Lightfield Rendering Small Part of Scene





Full Resolution Rendering: 500X Improvement!



Resolution Analysis

- ▶ Why do we have so much higher resolution in 2.0?
- ▶ Because the camera is focused:
 - ▶ The main lens creates radiance $r(x)$ at its image plane.
 - ▶ Plenoptic 1.0 and 2.0 sample this radiance differently.
- ▶ For one microcamera, the optical transfer matrix is A .
- ▶ Radiance on the sensor: $r'(x) = r(A^{-1}x)$
- ▶ (continue)

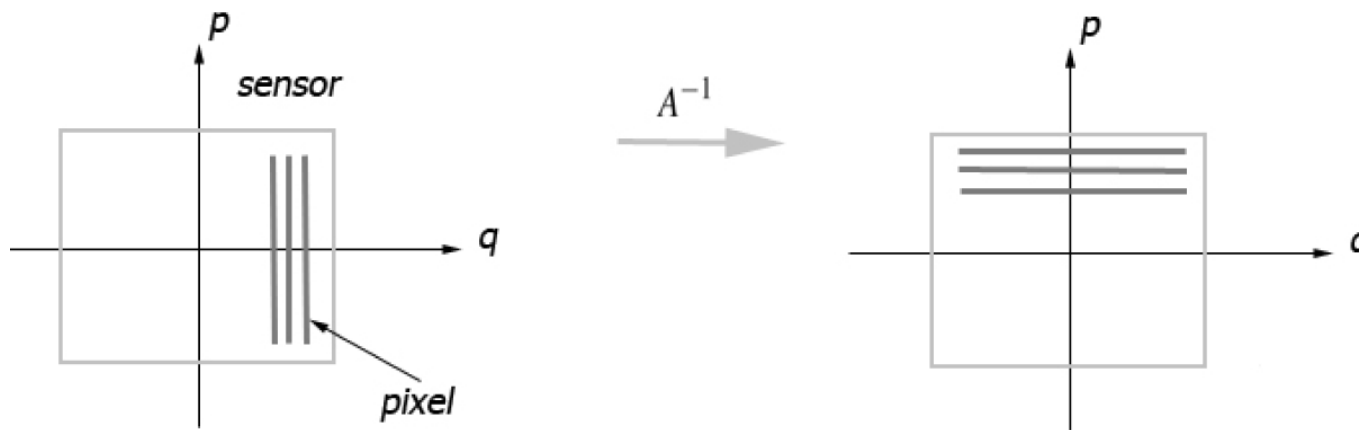


Resolution Analysis

- For Plenoptic 1.0 the transfer matrix is:

$$A = \begin{bmatrix} 1 & f \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix} = \begin{bmatrix} 0 & f \\ -\frac{1}{f} & 1 \end{bmatrix}$$

Inverse matrix $A^{-1} = \begin{bmatrix} 1 & -f \\ \frac{1}{f} & 0 \end{bmatrix}$



Resolution Analysis

- ▶ Rotation of each pixel to 90 degrees in optical phase space causes the low spatial resolution of 1.0 camera.
- ▶ For Plenoptic 2.0 the transfer matrix is:

$$A = \begin{bmatrix} 1 & b \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix} \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} -\frac{b}{a} & 0 \\ -\frac{1}{f} & -\frac{a}{b} \end{bmatrix}$$

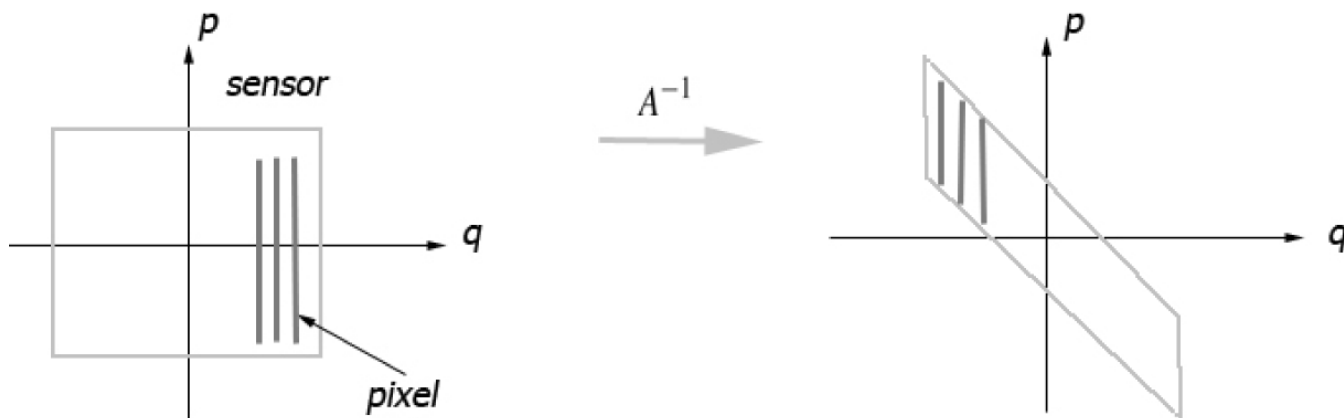
Inverse matrix $A^{-1} = \begin{bmatrix} -\frac{a}{b} & 0 \\ \frac{1}{f} & -\frac{b}{a} \end{bmatrix}$



Resolution analysis

- For Plenoptic 2.0 There is **no rotation**, just shear:

$$A^{-1} = \begin{bmatrix} -\frac{a}{b} & 0 \\ \frac{1}{f} & -\frac{b}{a} \end{bmatrix}$$

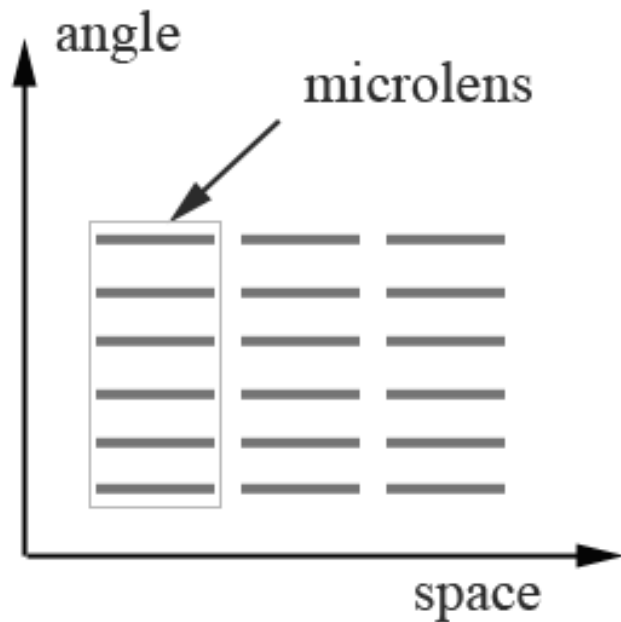


- Pixels remain “vertical”. b/a of the sensor resolution.

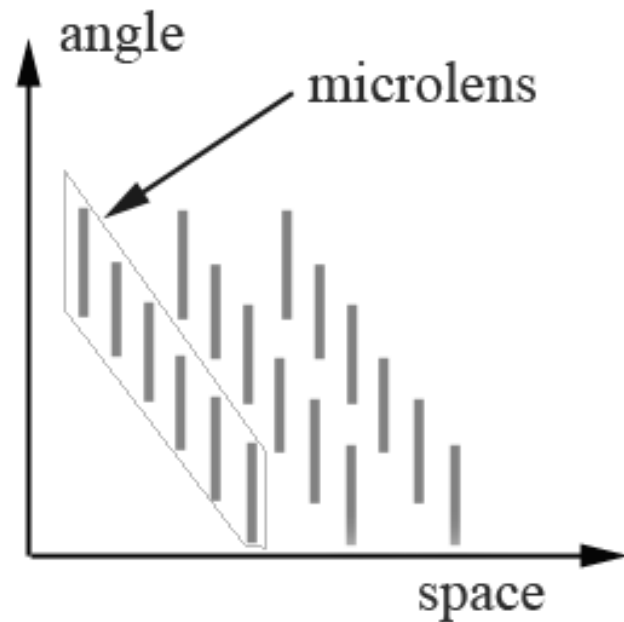


Resolution analysis

► Plenoptic 1.0



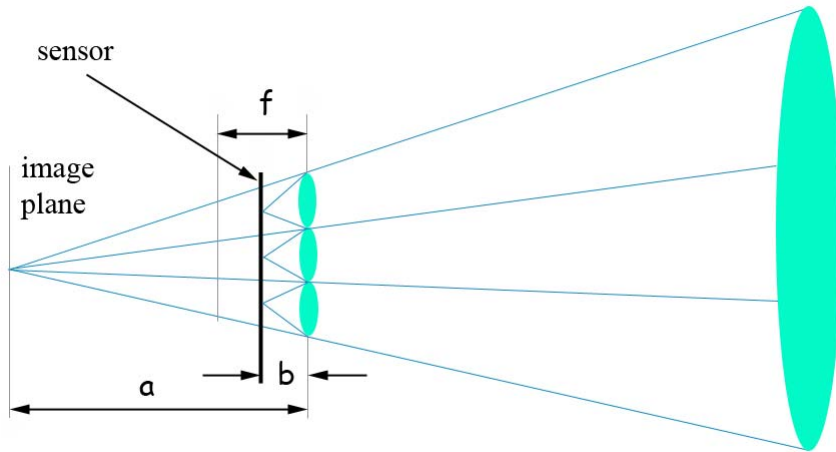
► Plenoptic 2.0



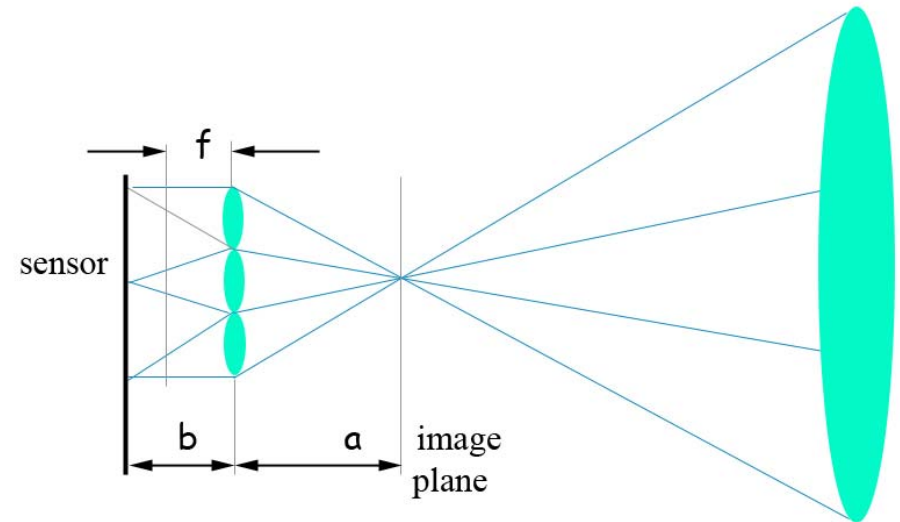


Two Ways of Focusing

► Galilean Telescopic Array



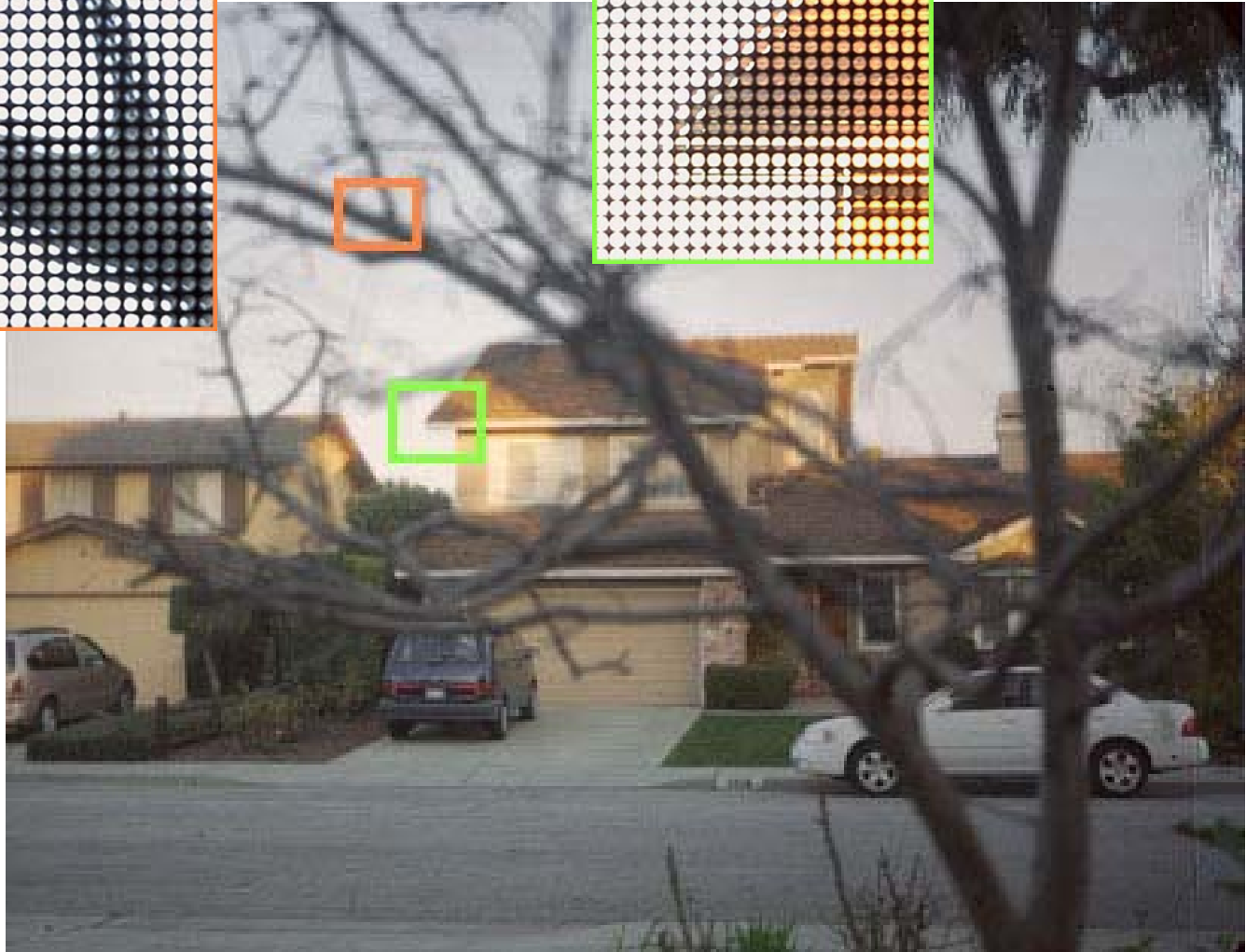
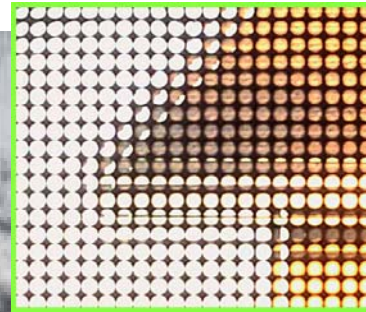
► Keplerian Telescopic Array



(Proposed by Galileo and Kepler 400 years ago)



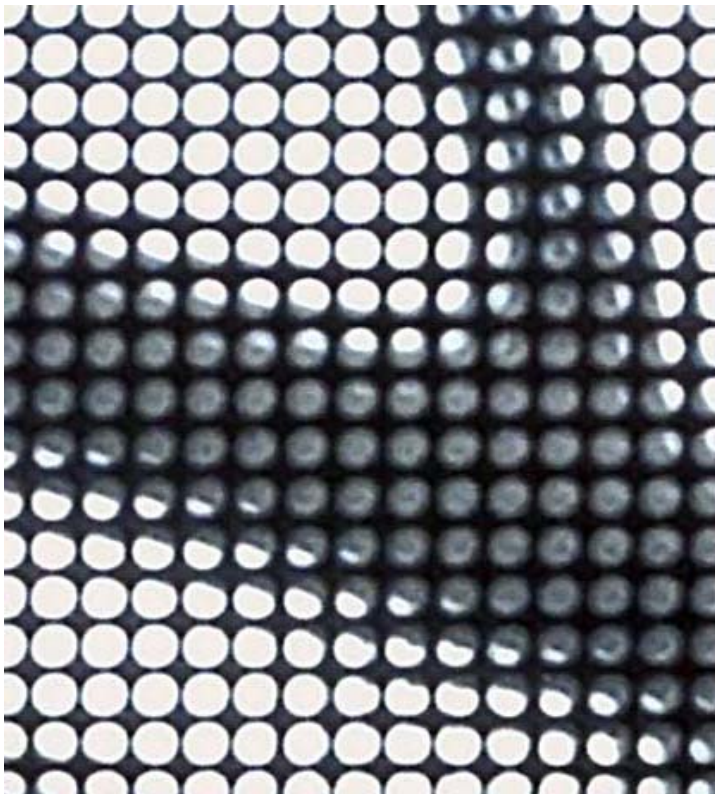
Two Ways of Focusing



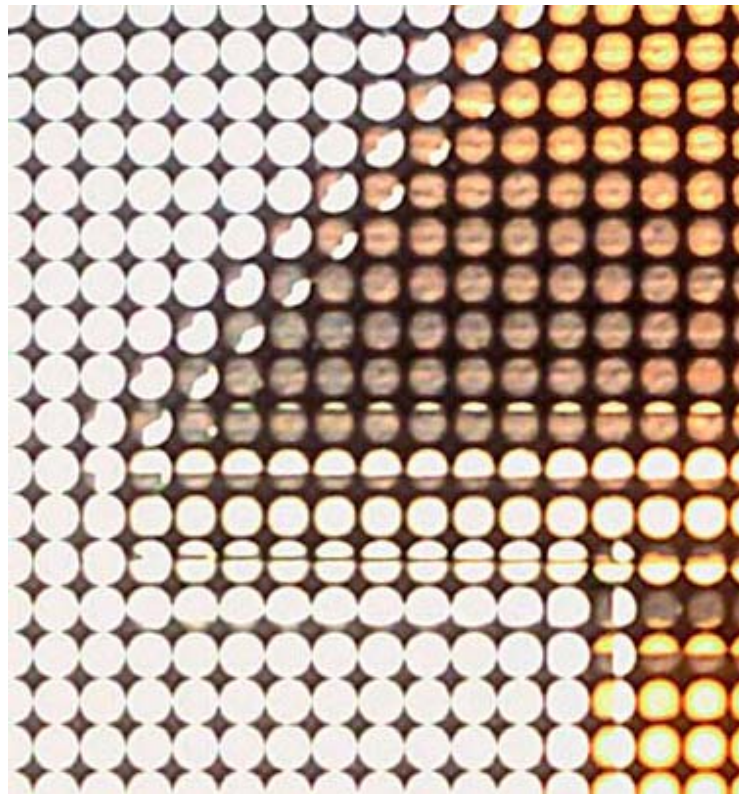


Two Ways of Focusing

► Galilean Imaging



► Keplerian Imaging





Plenoptic 2.0 Refocusing





Plenoptic 2.0 Refocusing





Plenoptic 2.0 Refocusing





Plenoptic 2.0 Resolution

- ▶ Plenoptic 2.0 sampling is more flexible:
 - ▶ Decouples resolution from number of microlenses.
 - ▶ Free to choose the spatial-angular tradeoff point.
- ▶ We can actually reach very low angular resolution not possible with traditional plenoptic camera (because edge effects would introduce noise).
 - ▶ Stereo 3D.
- ▶ Up to b/a of the sensor resolution can be achieved!
- ▶ This is up to 100%, i.e. full sensor resolution!

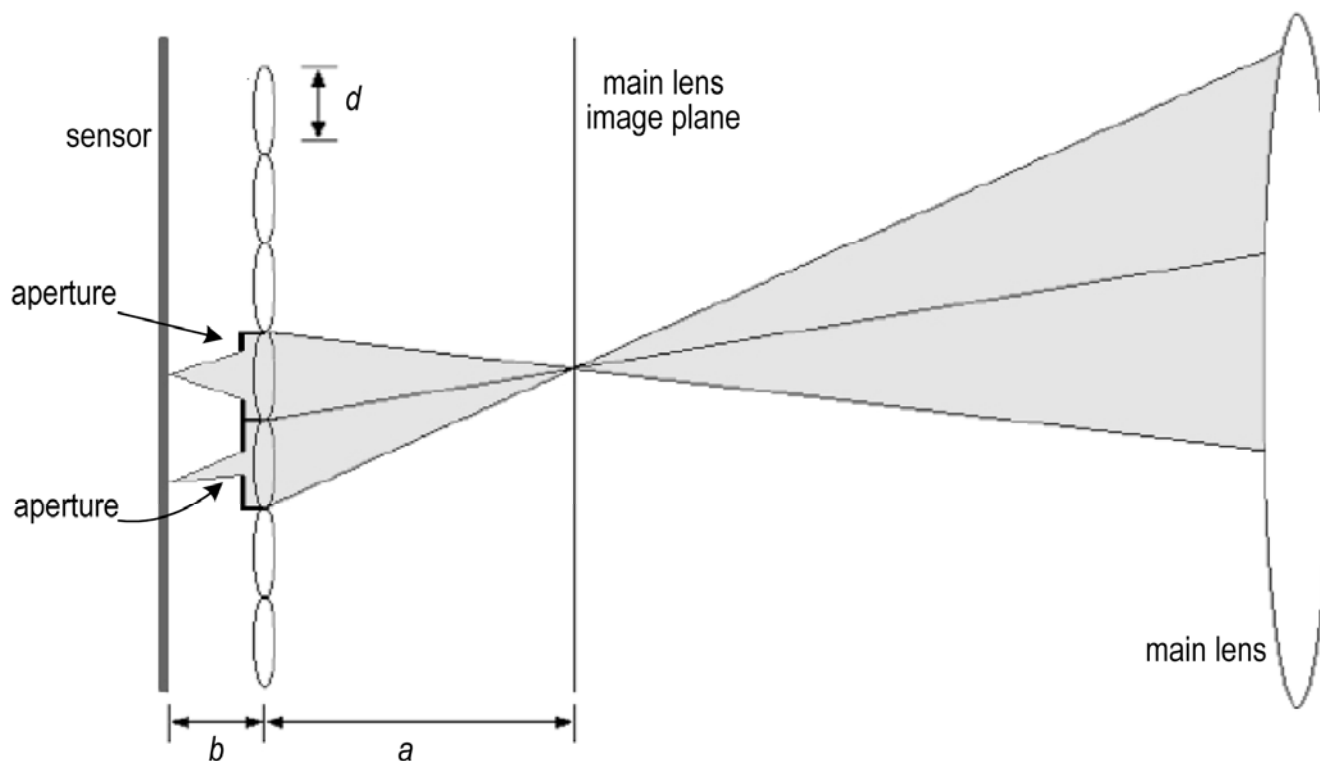


Plenoptic HDR Camera



HDR with Plenoptic Camera 2.0

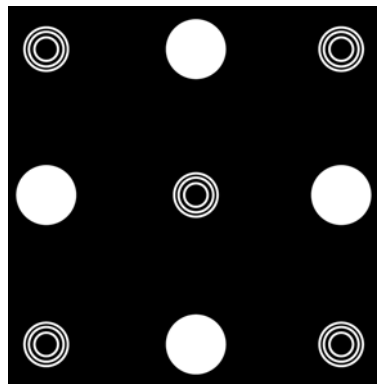
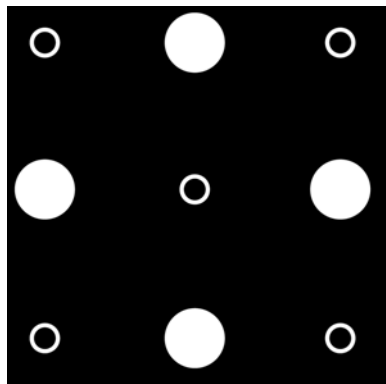
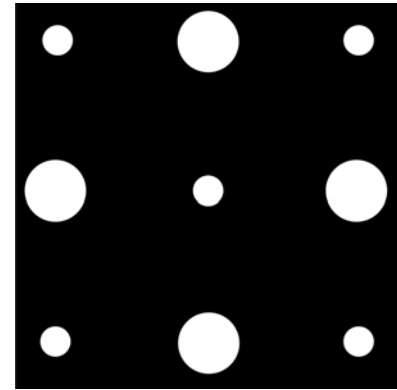
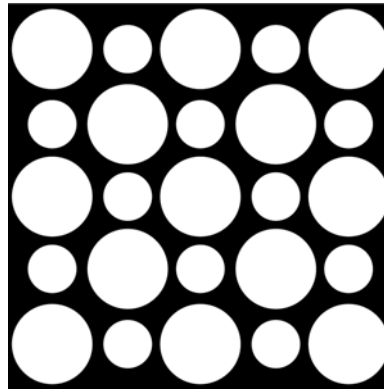
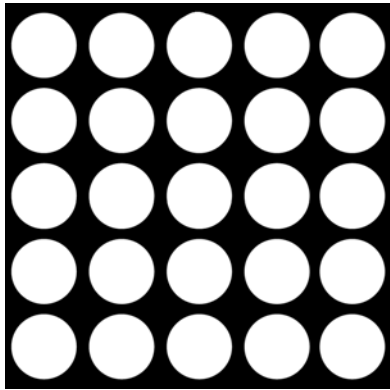
- ▶ Each point is seen multiple times in different microlenses
- ▶ We can put different apertures on different microlenses





HDR with Plenoptic Camera 2.0

- ▶ We can put different apertures on different microlenses

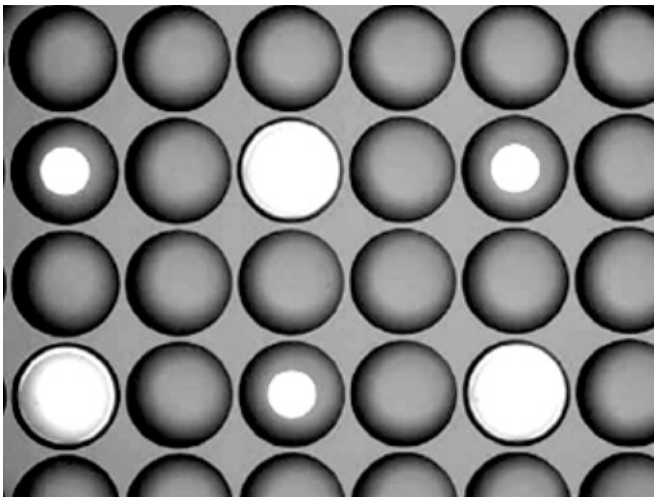


← Fresnel zones
as aperture

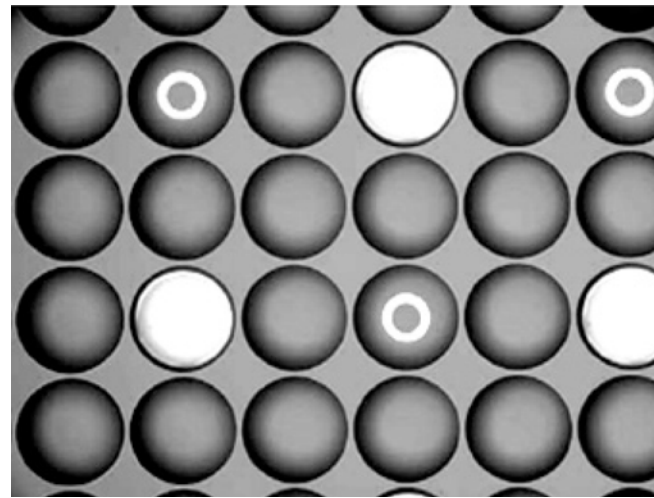


HDR with Plenoptic Camera 2.0

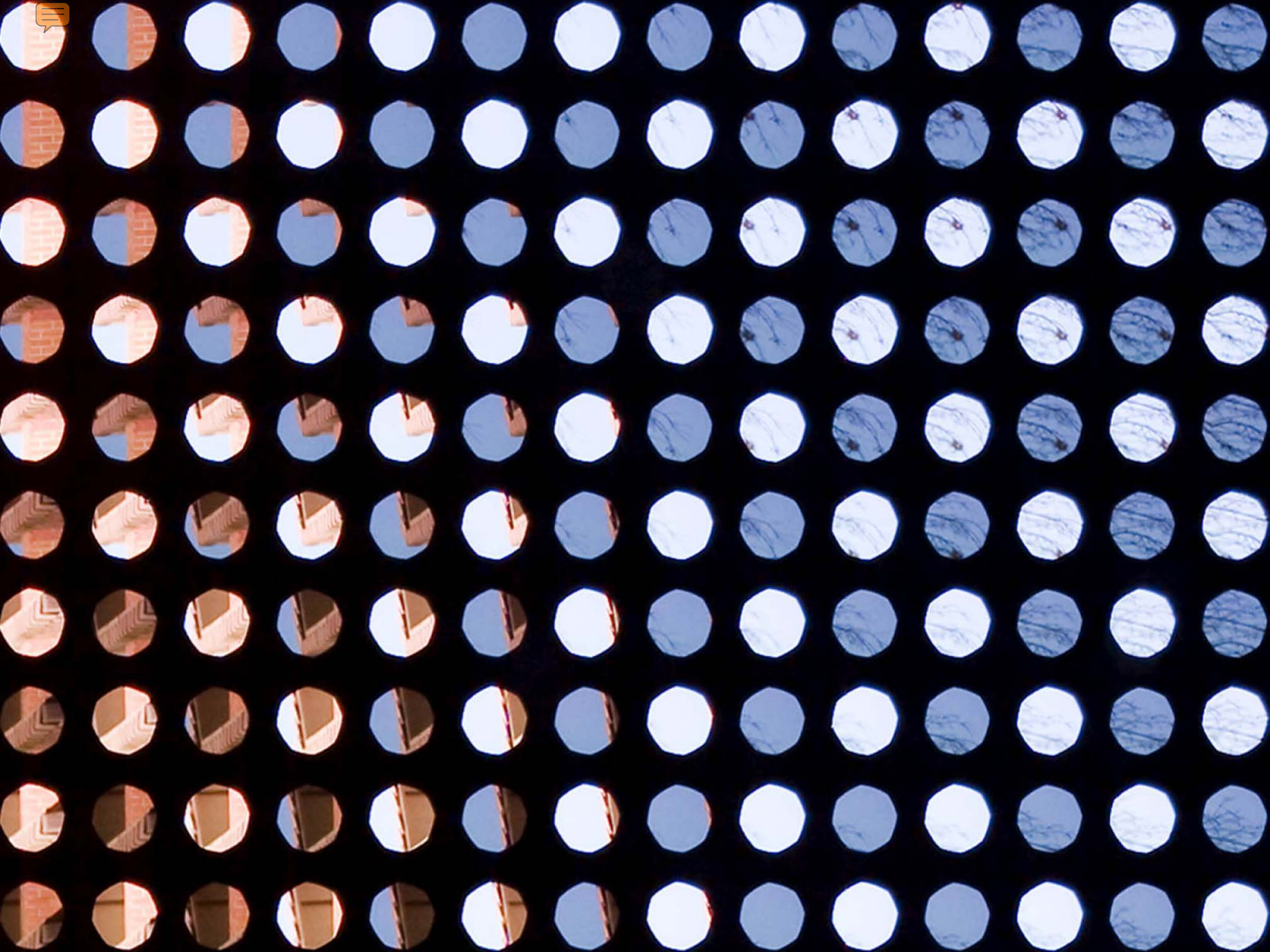
- ▶ Two of our microlens arrays under the microscope



4 times reduction
of aperture



8 times reduction
of aperture



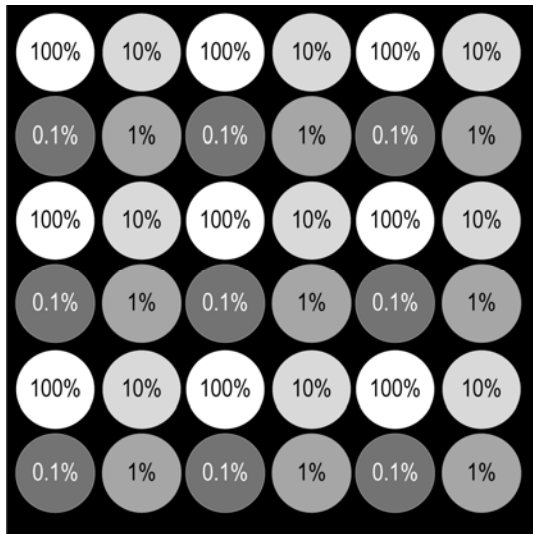




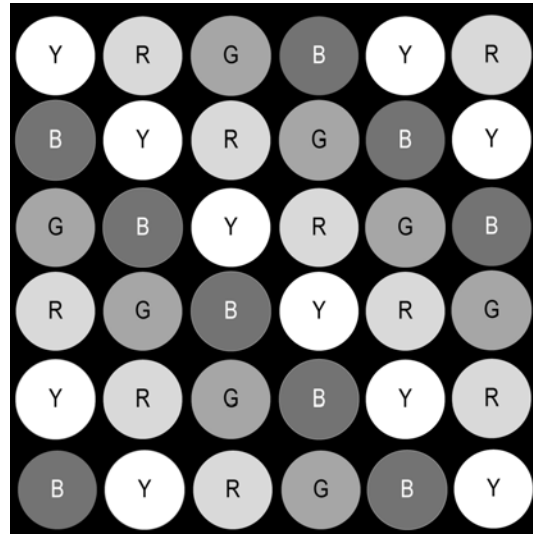


HDR with Plenoptic Camera 2.0

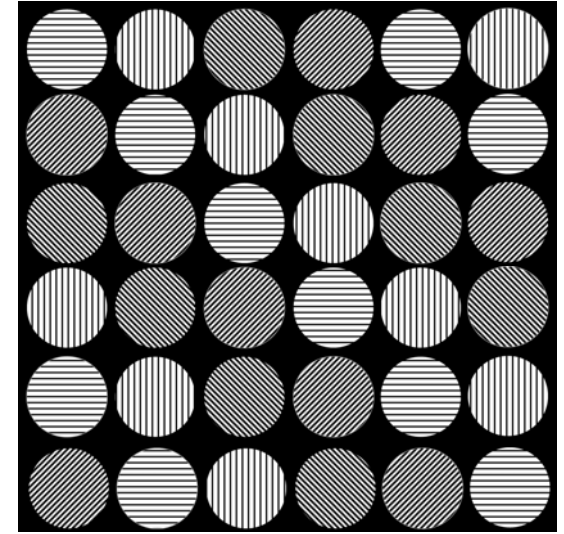
- ▶ We can put different filters on different microlenses: Neutral density, color (for spectral imaging), polarization.



1000 X increase
in dynamic range



12-color imaging
with RGB sensor



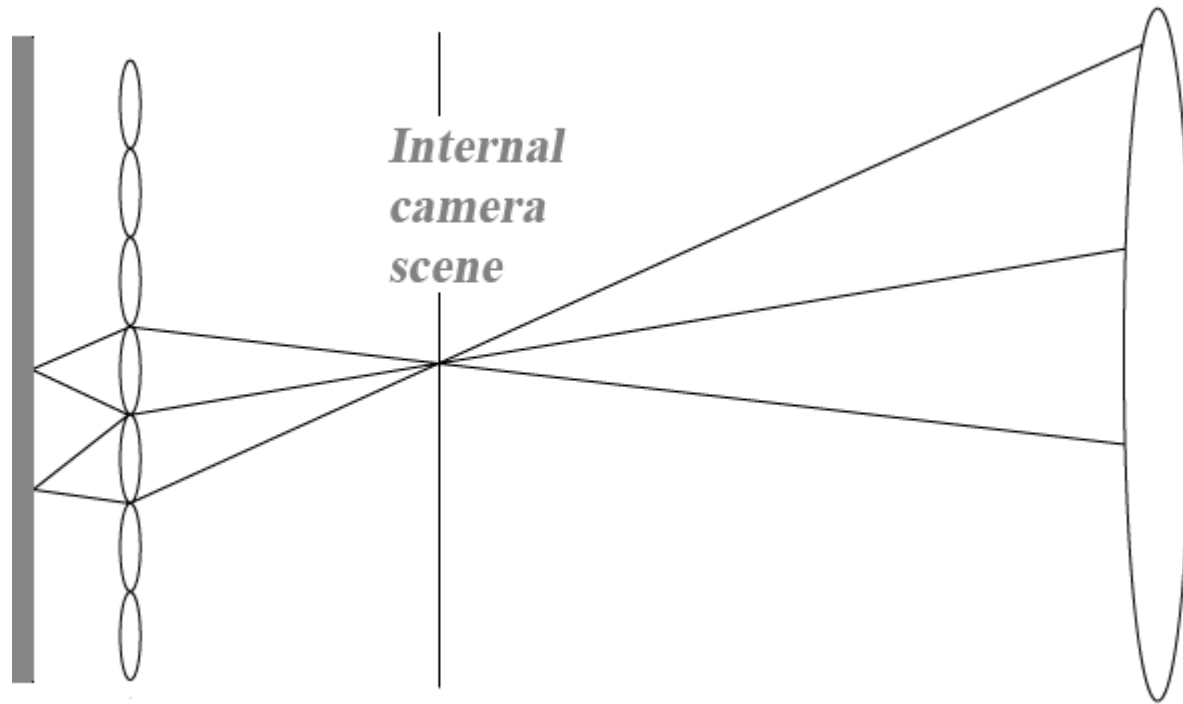
sampling four
linear polarizations

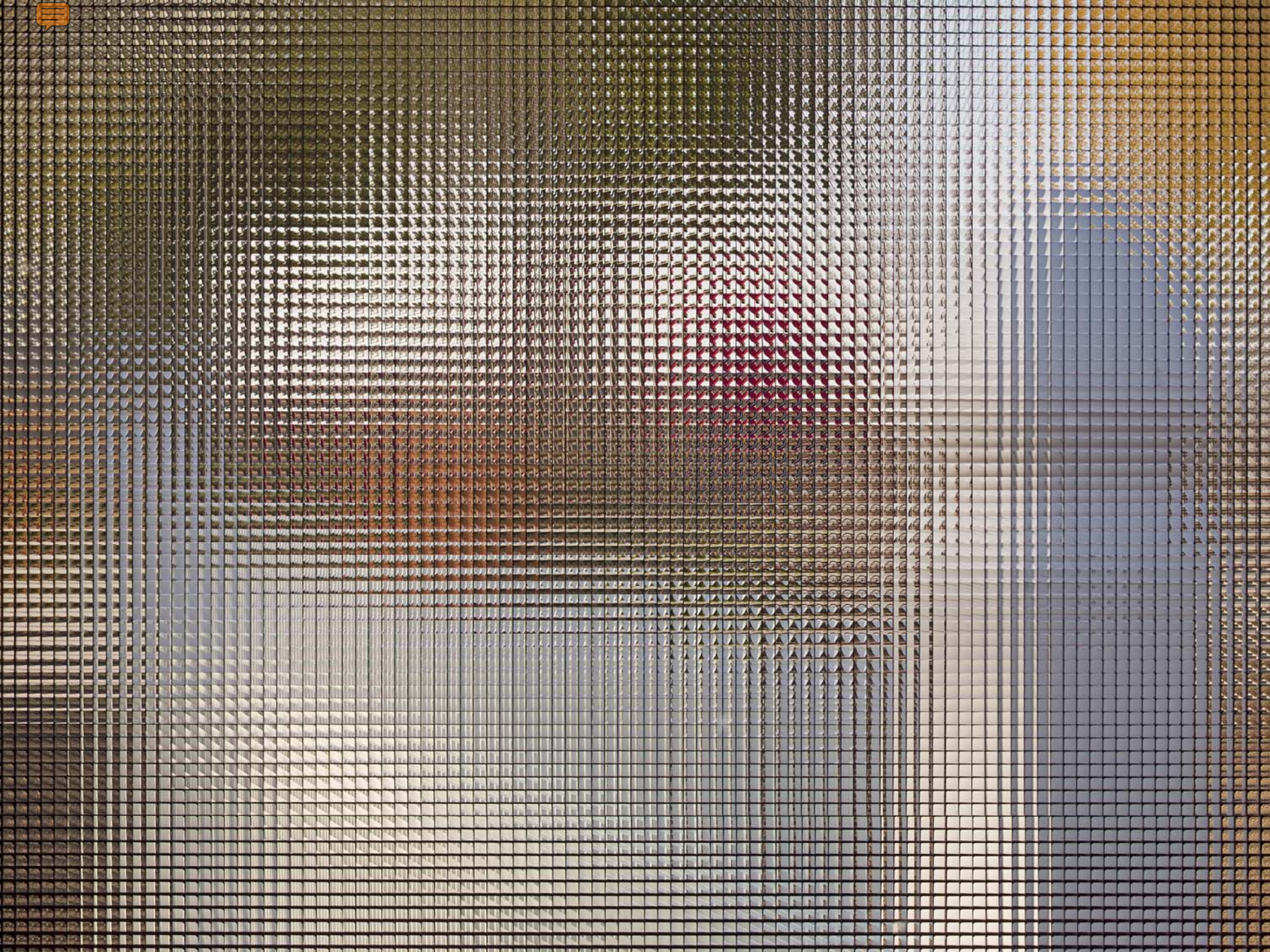


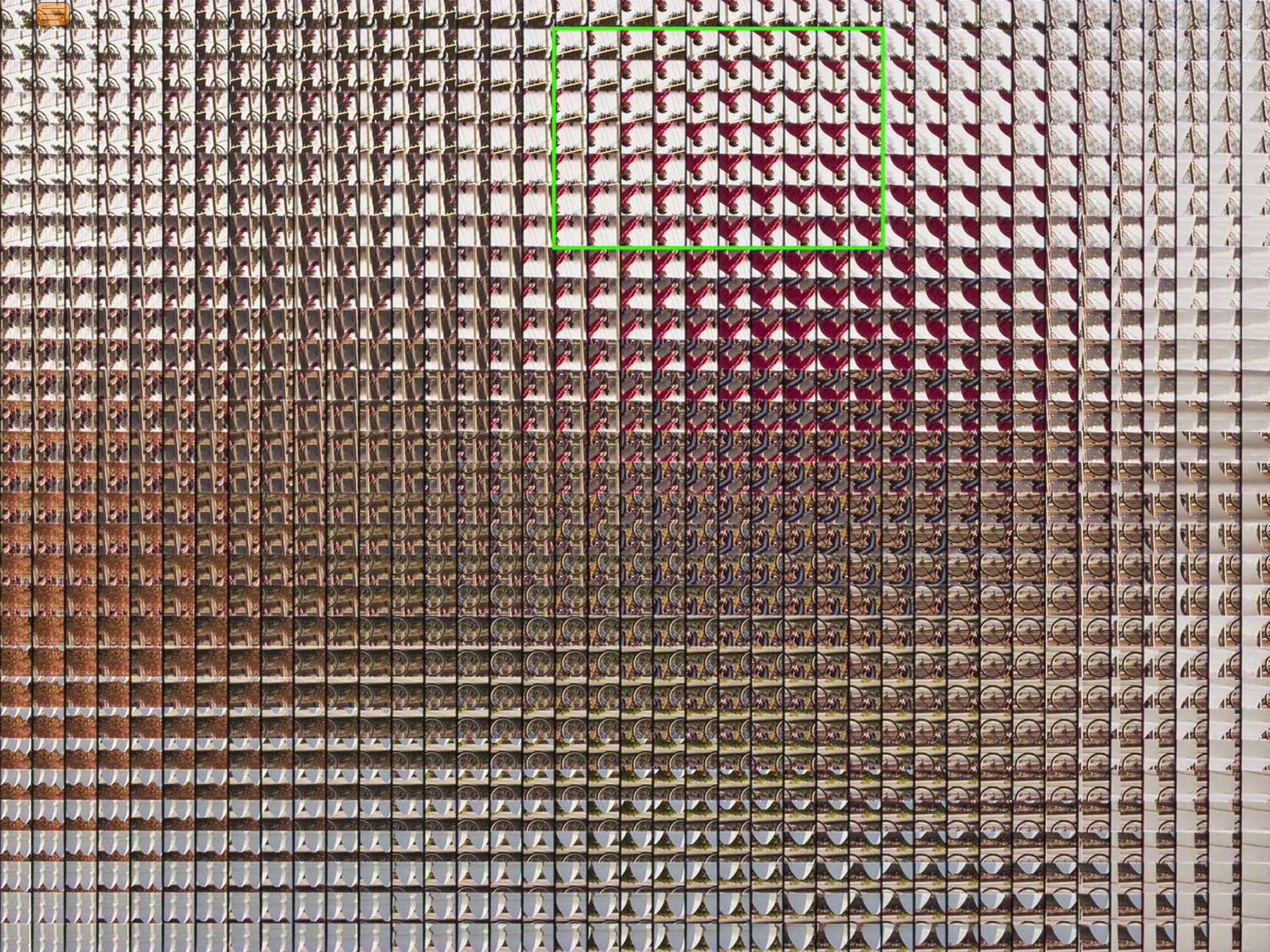
Plenoptic Superresolution Camera

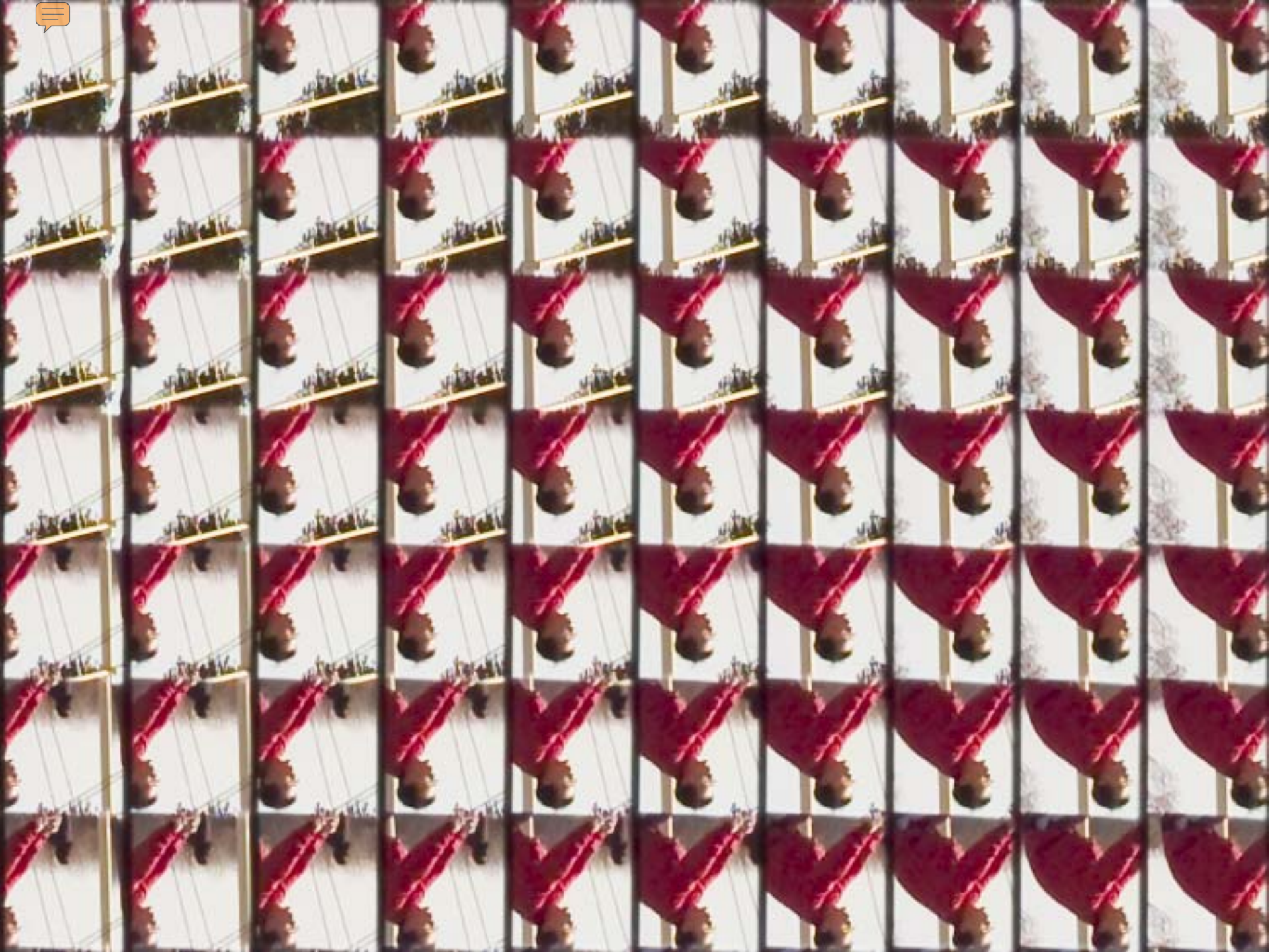
Superresolution with Plenoptic Camera 2.0

- ▶ Each microlens is observing the scene as a slightly shifted camera. We can compute the subpixel shift based on camera parameters. Then, superresolve.



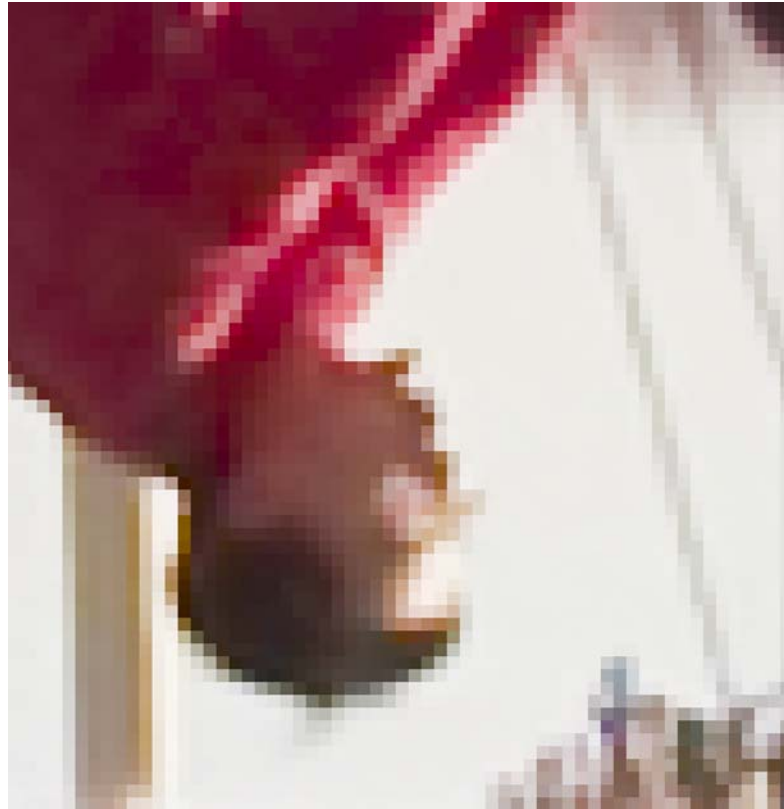








Superresolution with Plenoptic Camera 2.0



- Observe the subpixel shift



Superresolution with Plenoptic Camera 2.0



- Observe the subpixel shift







Superresolution with Plenoptic Camera 2.0

- ▶ Compare same image with traditional lifgtfield rendering (plenoptic 1.0).









Hands On with Radiance Cameras



Handheld Plenoptic Camera



Computational Methods for Radiance

Render “the full variety offered by the direct observation of objects.”

Methods for Plenoptic 1.0



Computing with Radiance

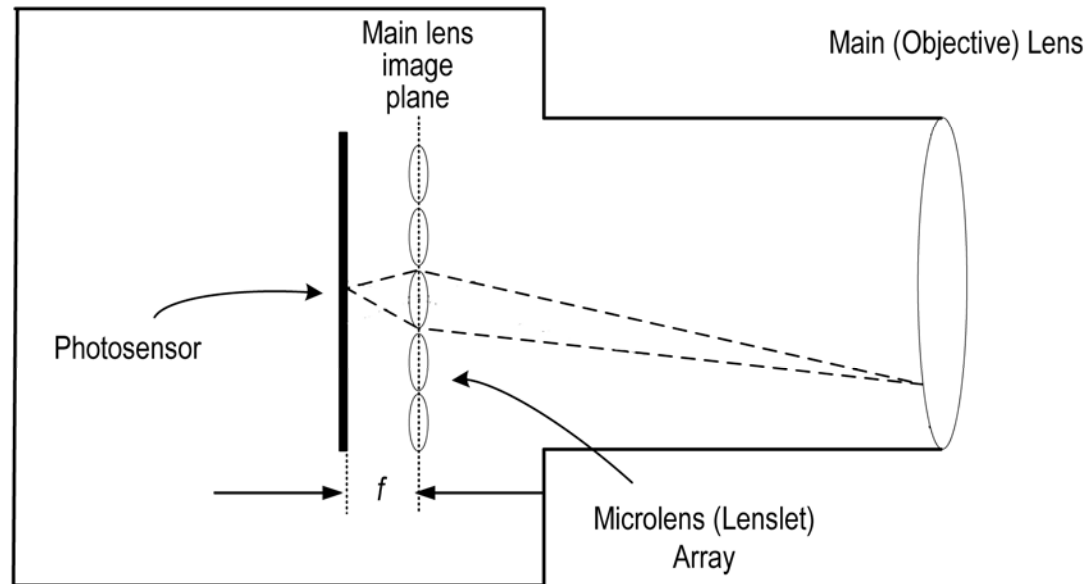
- ▶ Goal: Render “the full variety offered by the direct observation of objects.”
- ▶ Computational tasks:
 - ▶ Interpreting a digital plenoptic image as radiance
 - ▶ Rendering radiance to image
 - ▶ Algorithms for transforming radiance
 - ▶ Real-time interactive implementation using GPUs



Radiance Representation (Plenoptic 1.0)

- ▶ Sensor image represents sampled radiance
 - ▶ Position is sampled by microlenses “as pixels”

Conventional plenoptic camera



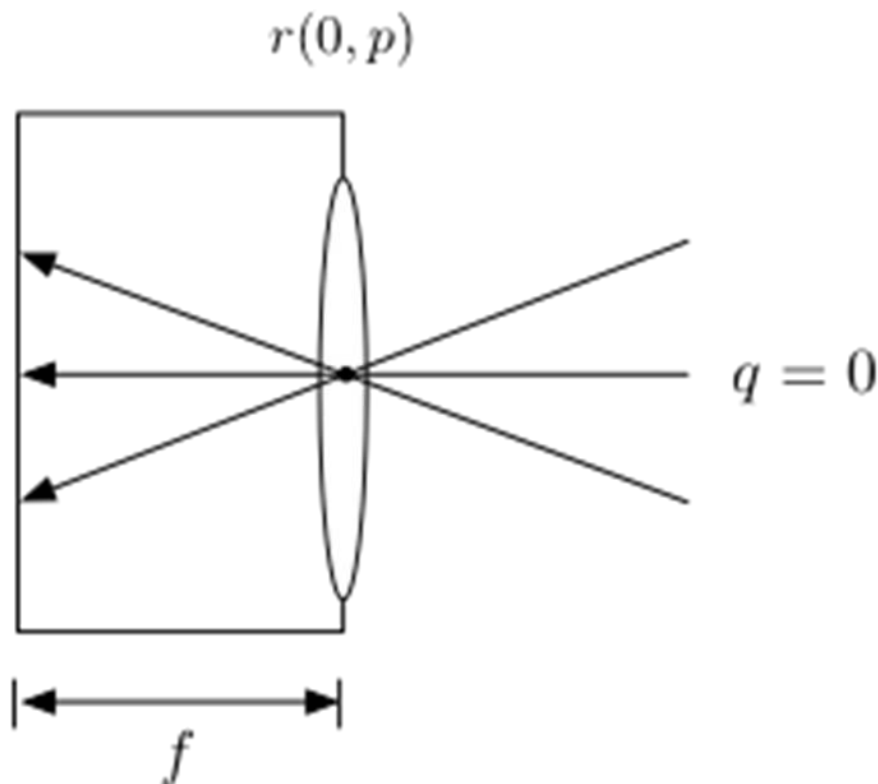


Radiance Representation (Plenoptic 1.0)

- ▶ Sensor image represents sampled radiance

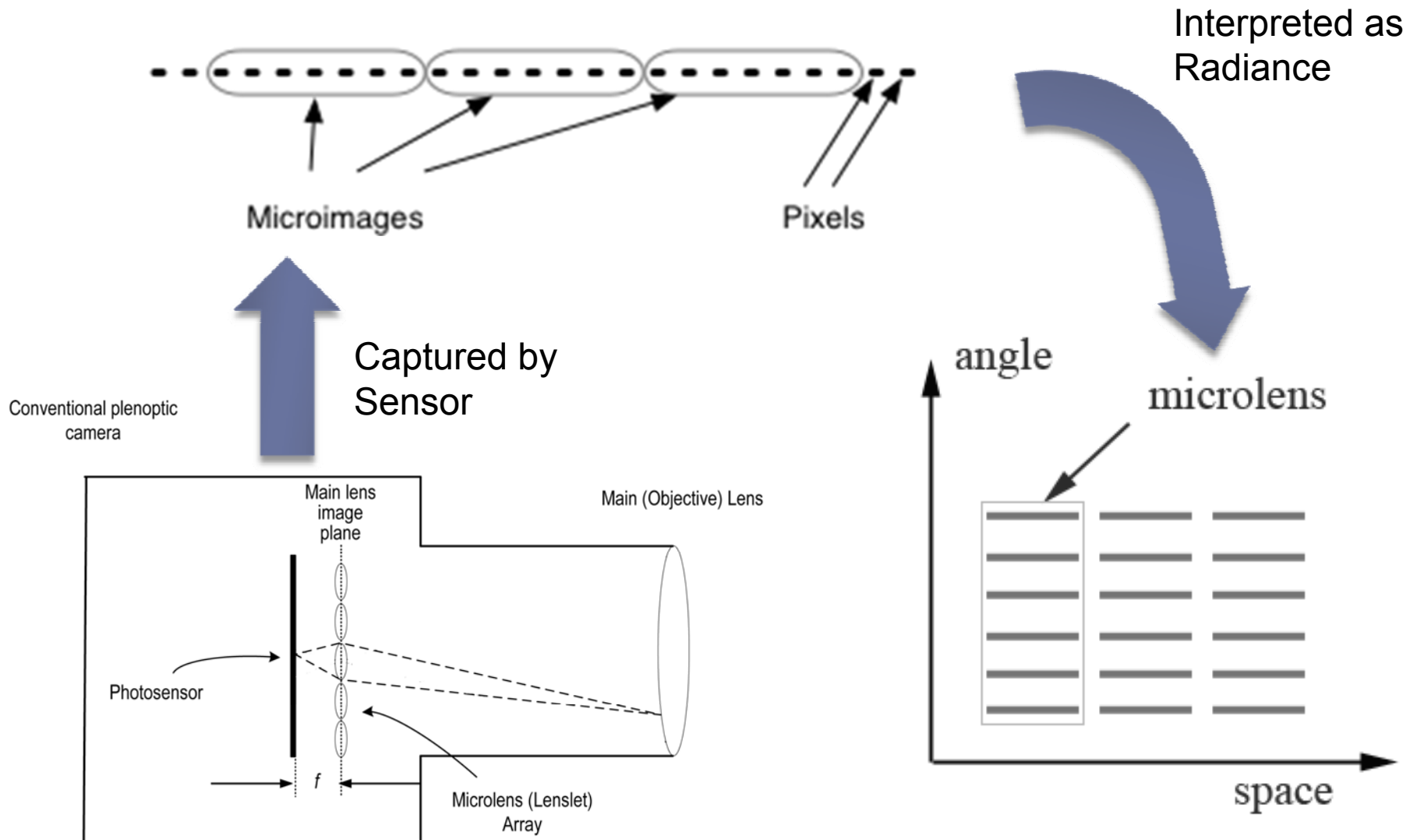
- ▶ Position is sampled by microlenses
- ▶ Direction is sampled by sensor pixels

$$I(q) = \frac{d}{f} r(0, \frac{q}{f})$$



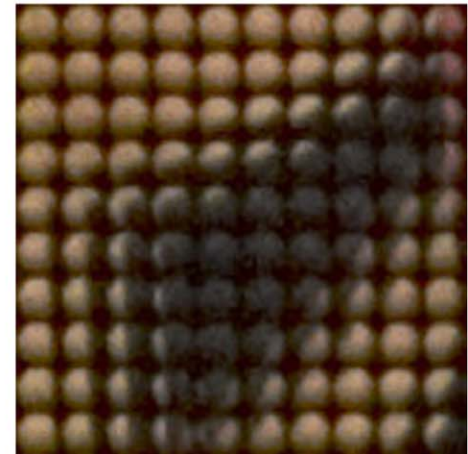
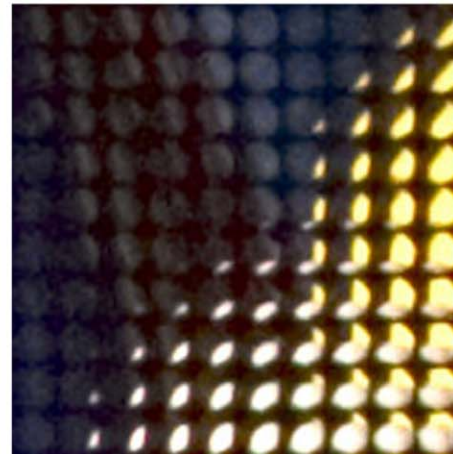
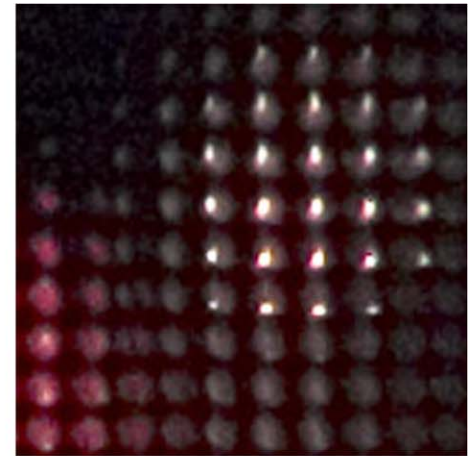
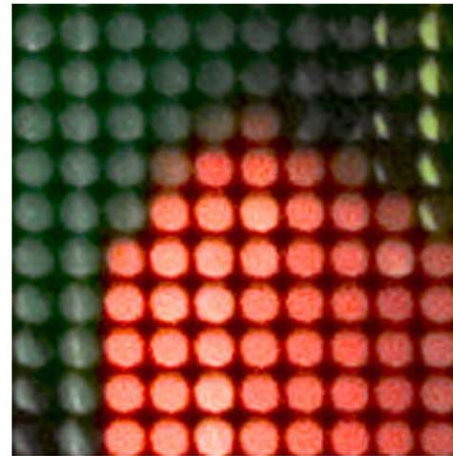
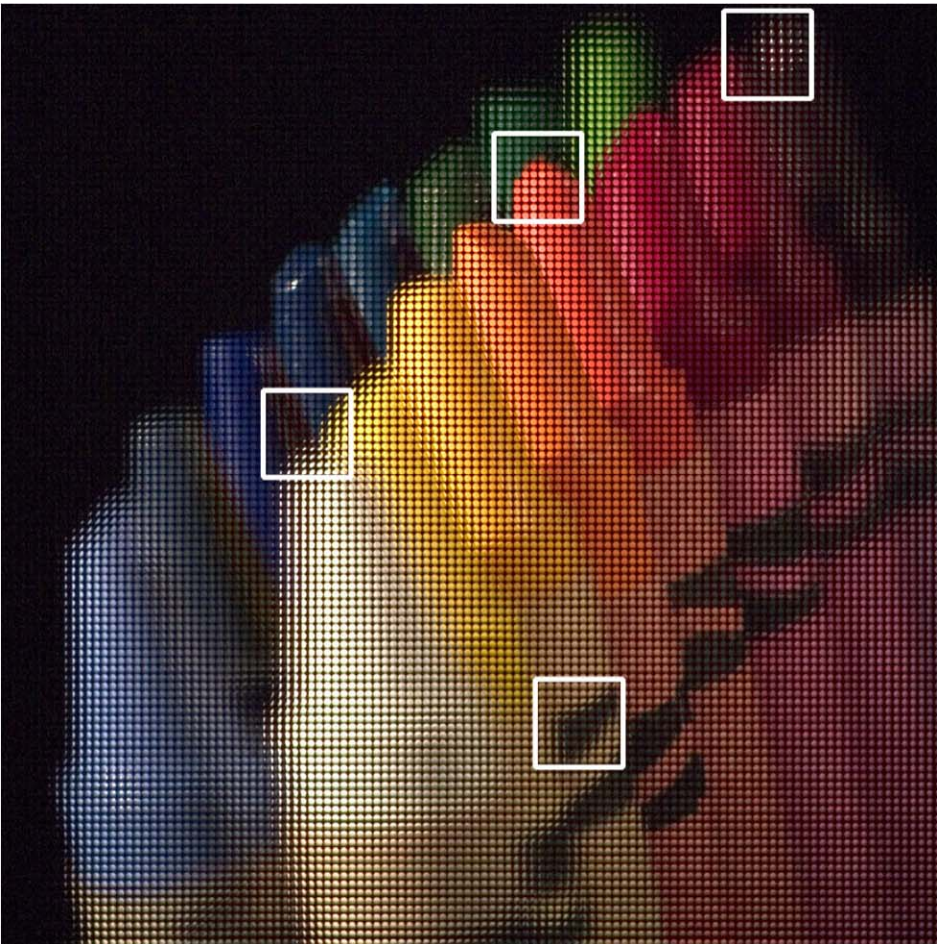


Radiance Representation (Plenoptic 1.0)





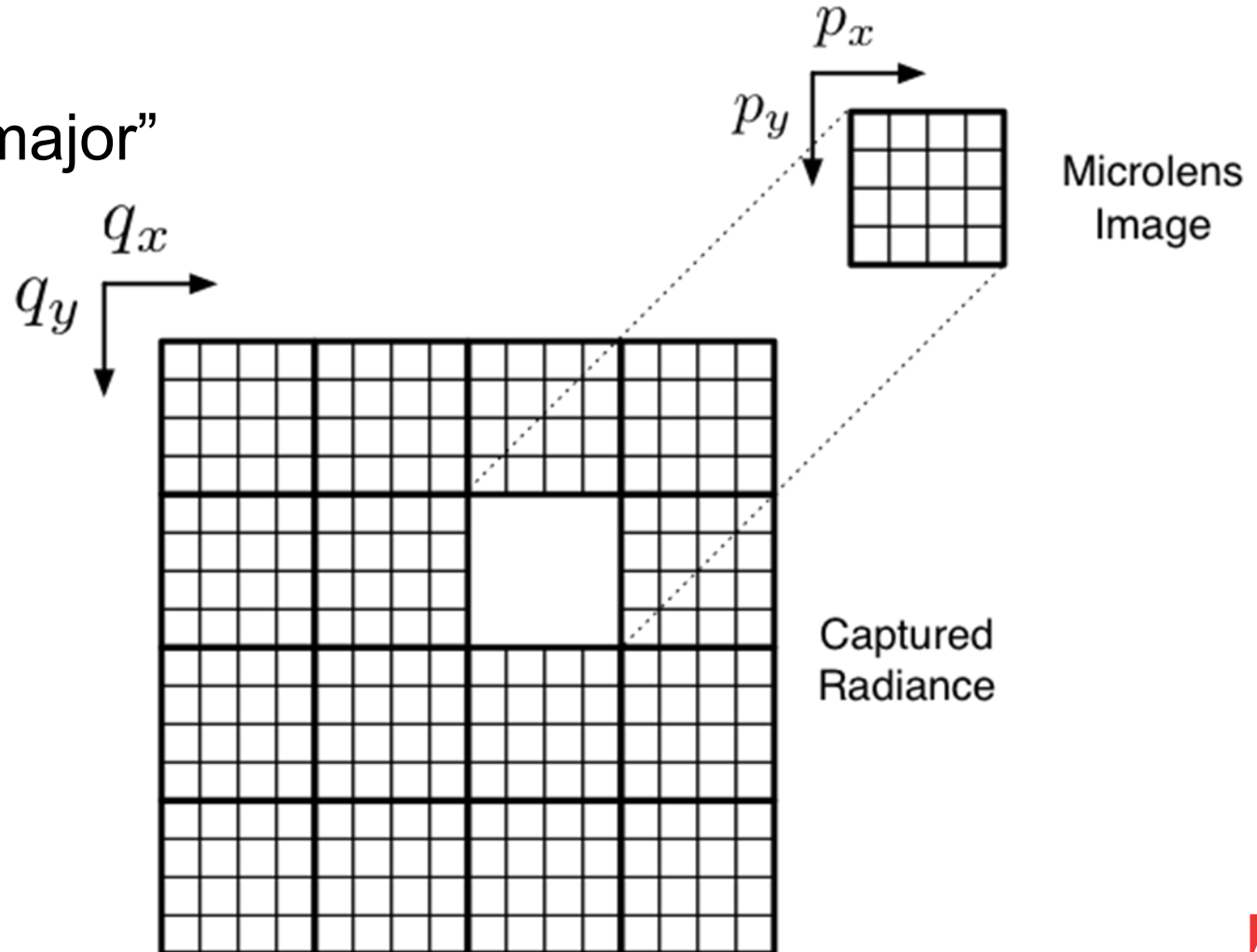
Radiance Representation (Plenoptic 1.0)





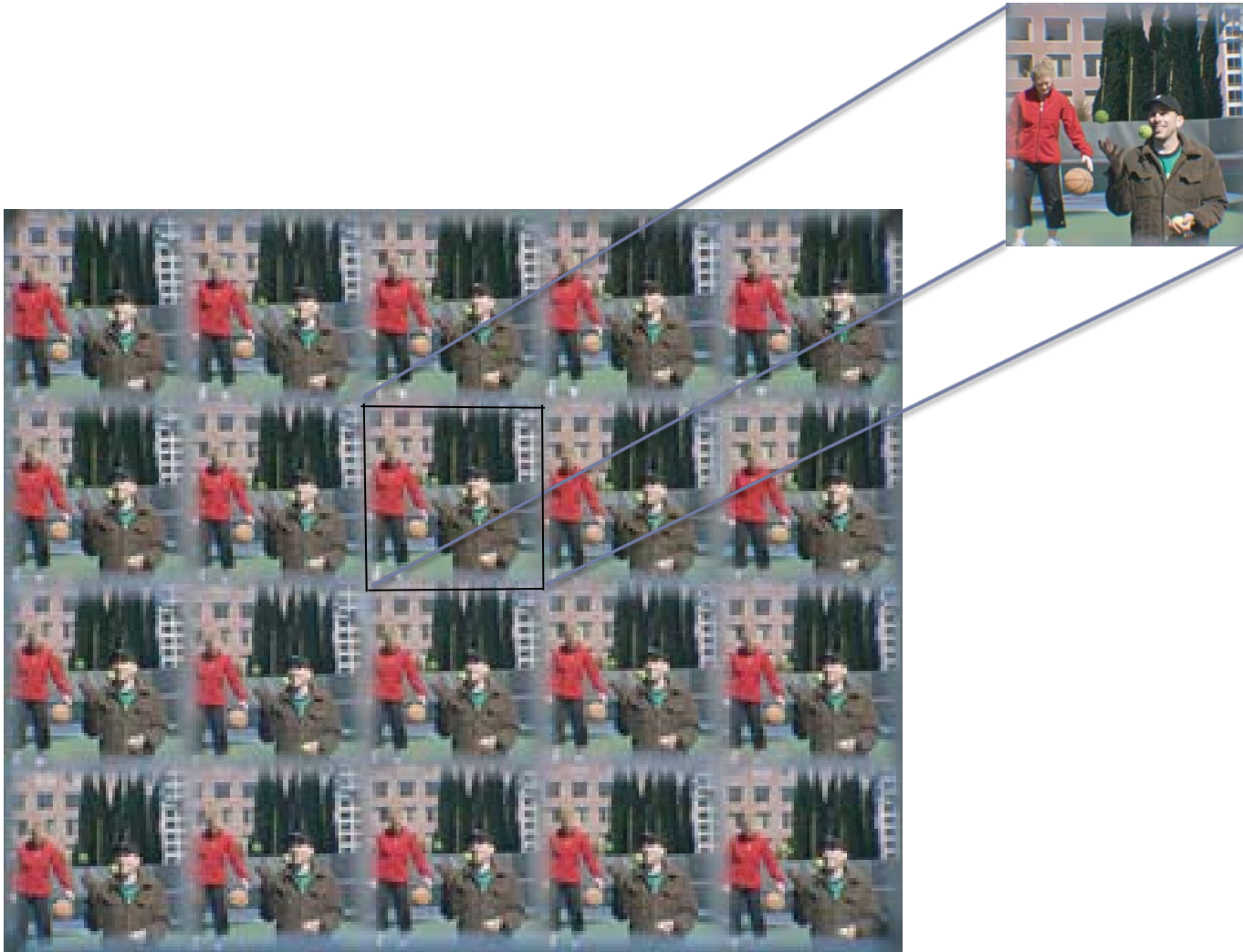
Radiance Representation (Plenoptic 1.0)

- ▶ Plenoptic image is a “flat” 2D array of 2D arrays
- ▶ 4D array
- ▶ “Position major”





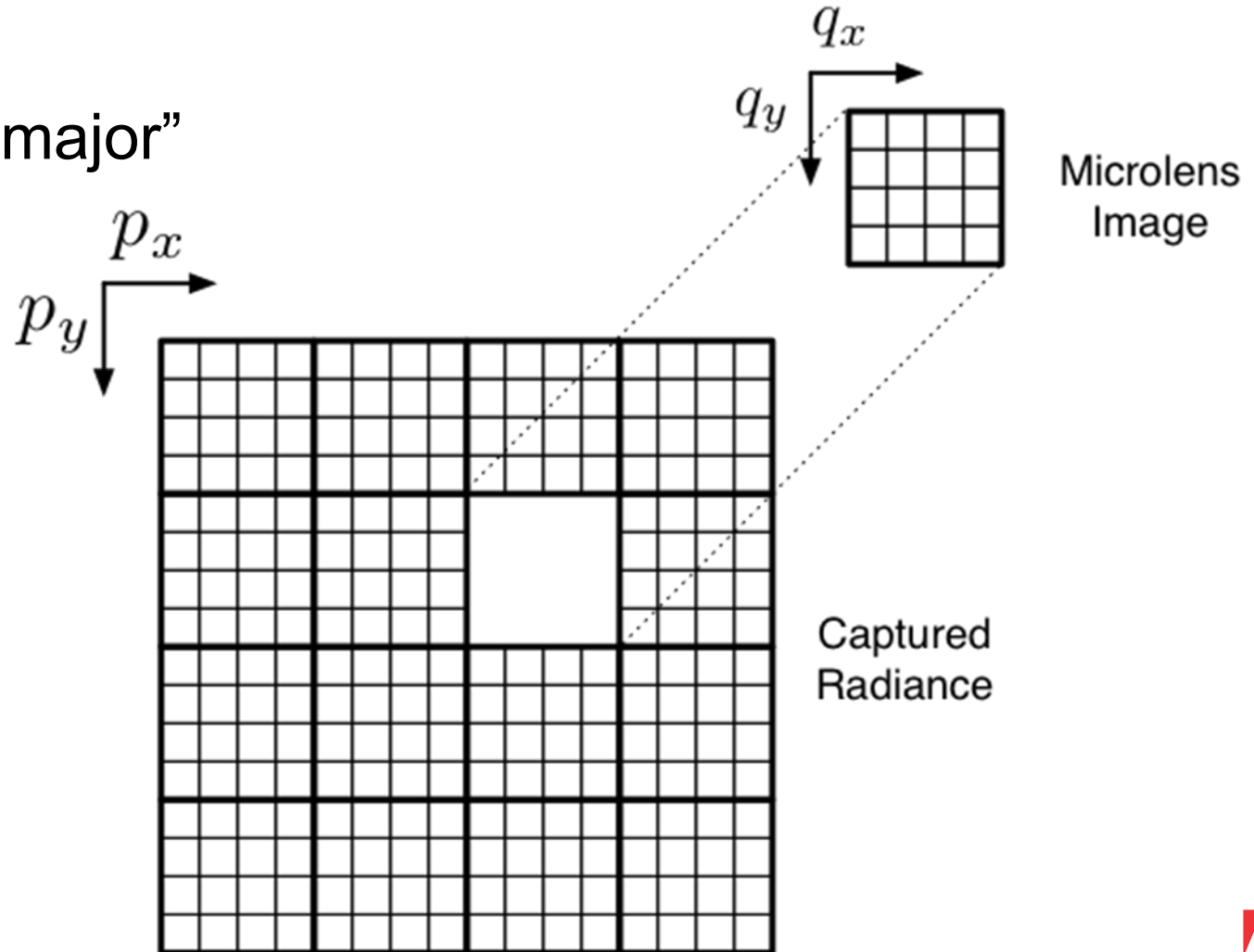
Radiance Representation (Plenoptic 1.0)





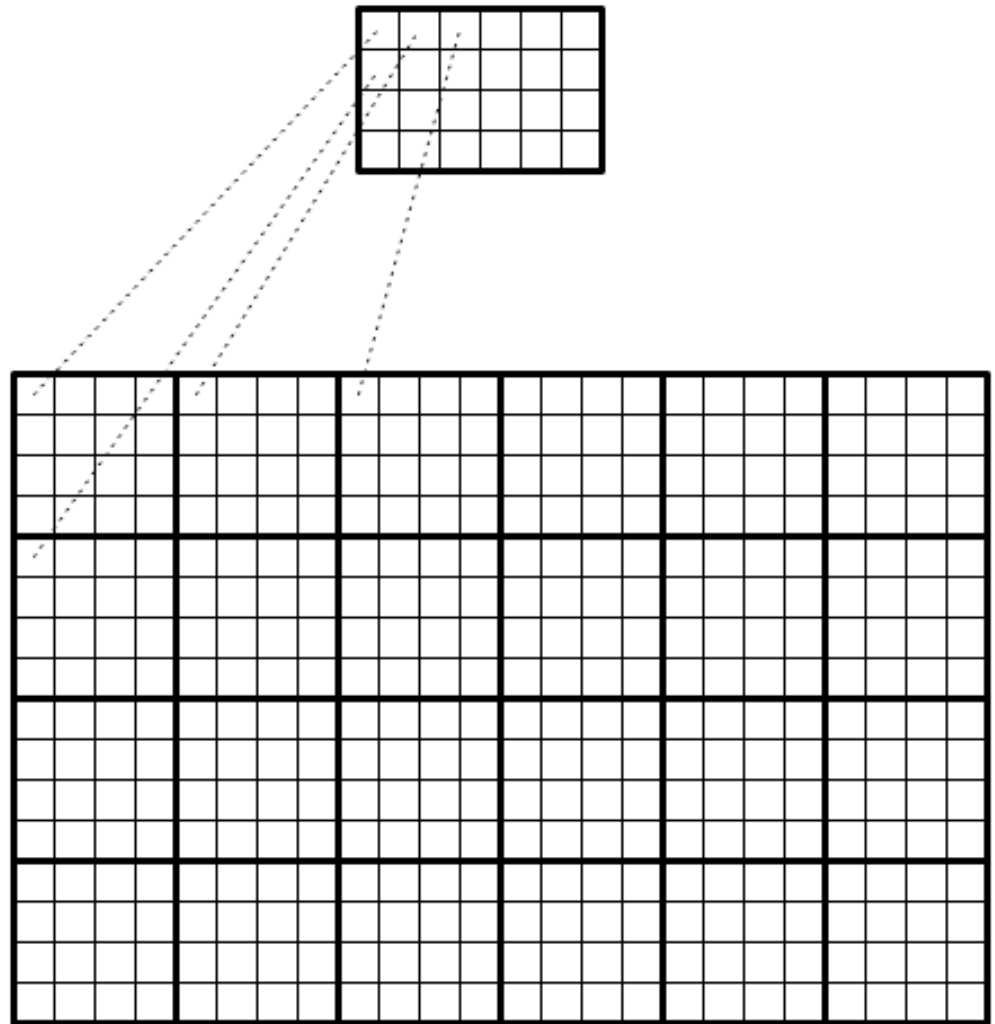
Radiance Representation (Plenoptic 1.0)

- ▶ Plenoptic image is a “flat” 2D array of 2D arrays
- ▶ 4D array
- ▶ “Direction major”



Creating Radiance Array (Position Major)

- ▶ Given 2D “flat” captured by radiance camera
- ▶ Create 4D array
- ▶ Sample the same directional pixel from every position
- ▶ Convention $r[i, j, m, n]$
 - ▶ Follow $r(q, p)$
 - ▶ i, j are position
 - ▶ m, n are direction



Creating Radiance Array (Position Major)

- ▶ Given 2D position major “flat” from radiance camera
- ▶ Create 4D array
- ▶ If 2D position major “flat” is regular

```
(jnds,inds) = mgrid[0:height:nump,0:width:nump]
```

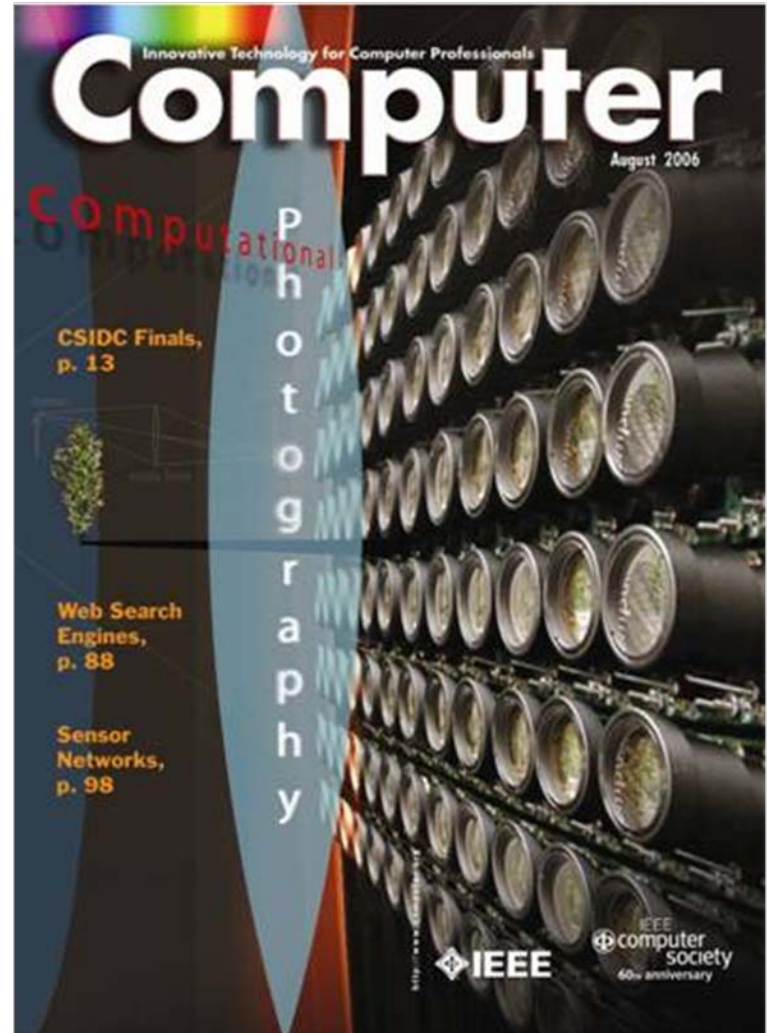
```
for j in range(0,nump):  
    for i in range(0,nump):  
        radiance[:, :, j, i] = image[jnds+j,inds+i]
```

- ▶ Python, matlab very similar
- ▶ Samples the same directional pixel from every position



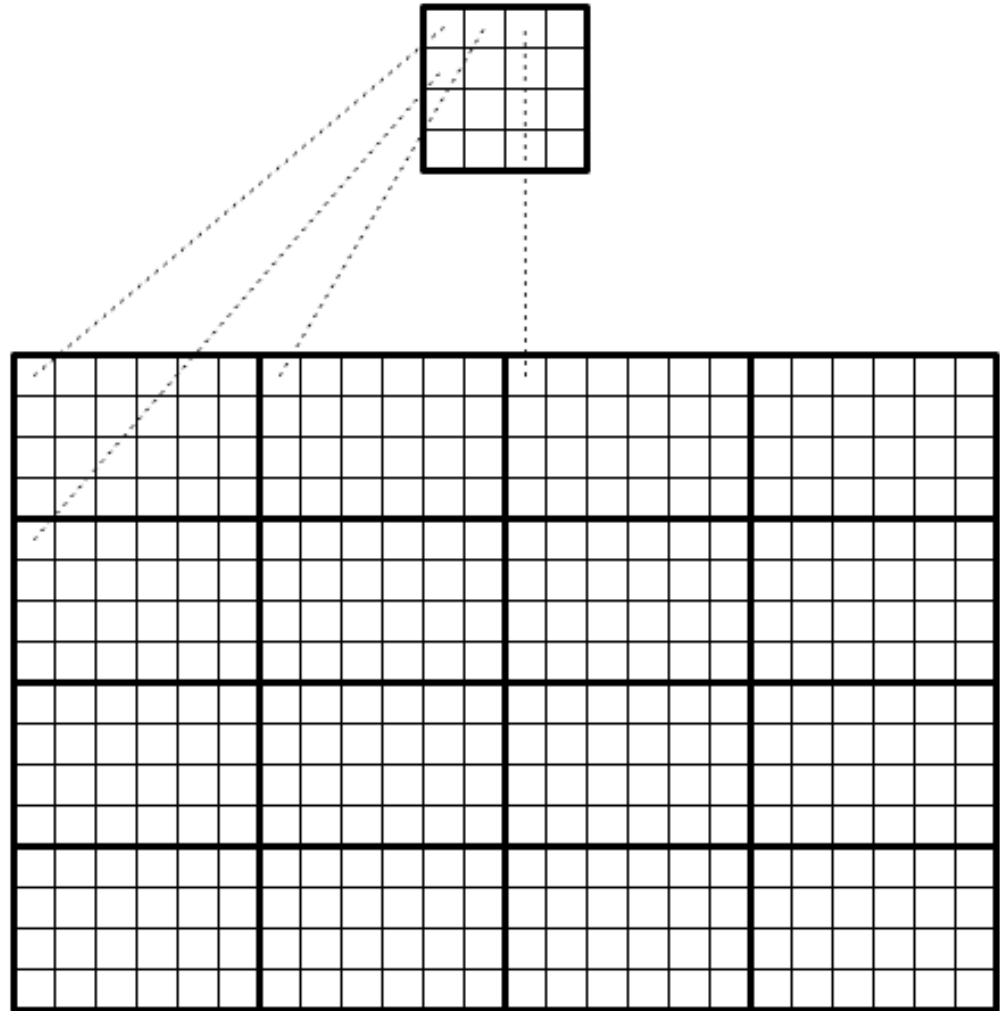
Camera Arrays

- ▶ The most popular lightfield camera is simply an array of conventional cameras, like the Stanford array.
- ▶ Alternatively, an array of lenses/prisms with a common sensor, like the Adobe array.



Creating Radiance Array (Direction Major)

- ▶ Given 2D “flat” captured by radiance camera
- ▶ Create 4D array
- ▶ Sample the same positional pixel from every direction
- ▶ Convention $r[i, j, m, n]$
 - ▶ Follow $r(q, p)$
 - ▶ i, j are position
 - ▶ m, n are direction



Creating Radiance Array (Direction Major)

- ▶ Given 2D direction major “flat” from radiance camera
- ▶ Create 4D array
- ▶ If 2D direction major “flat” is regular

```
(jnds,inds) = mgrid[0:height:numq,0:width:numq]
```

```
for j in range(0,numq):
```

```
    for i in range(0,numq):
```

```
        radiance[j,l,:,:) = image[jnds+j,inds+i]
```

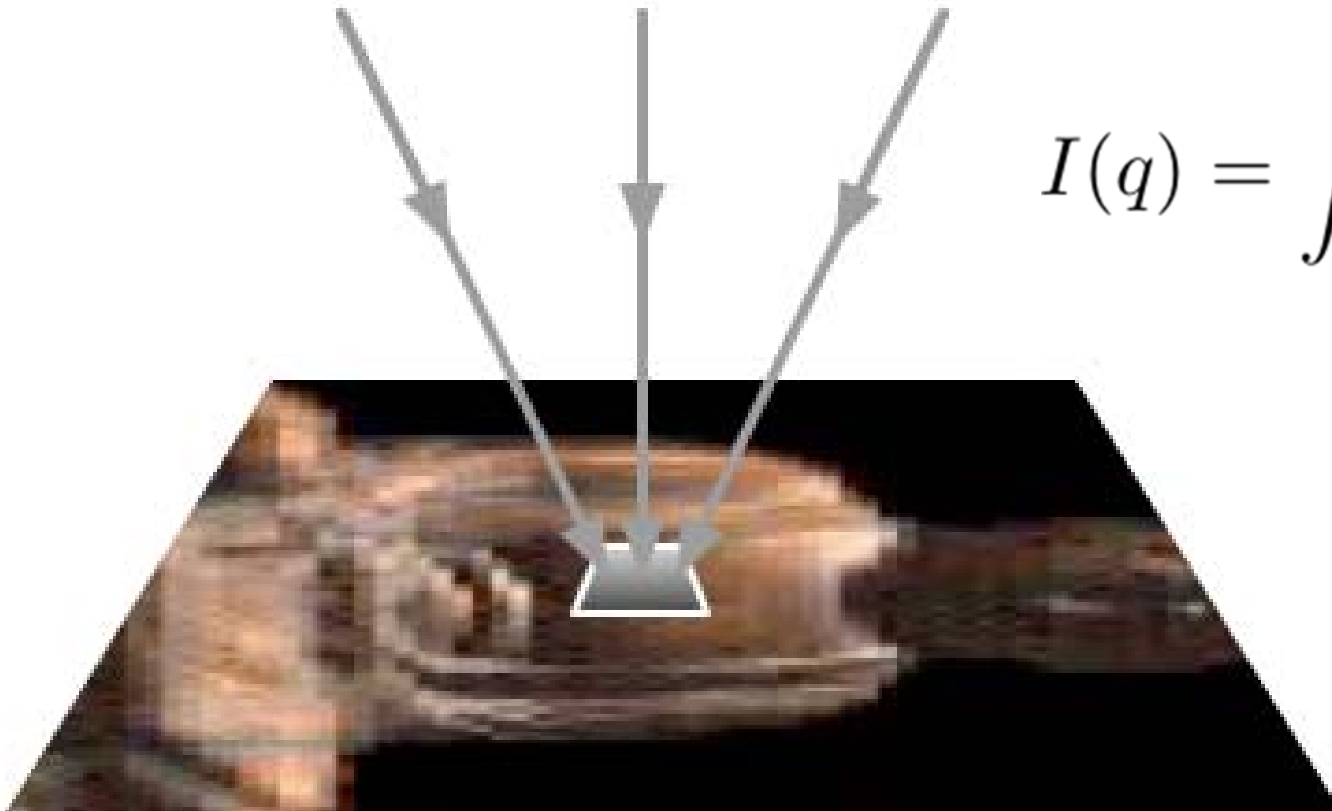
- ▶ Python, matlab very similar
- ▶ Samples the same positional pixel from every direction

Aside: Dimensionality

- ▶ How large of a sensor do we need to capture radiance?
- ▶ Memory, computation requirements?
- ▶ What is a reasonable size for a rendered image?

Image Rendering

- ▶ A traditional image (a picture) is formed by integrating rays from every direction at each pixel

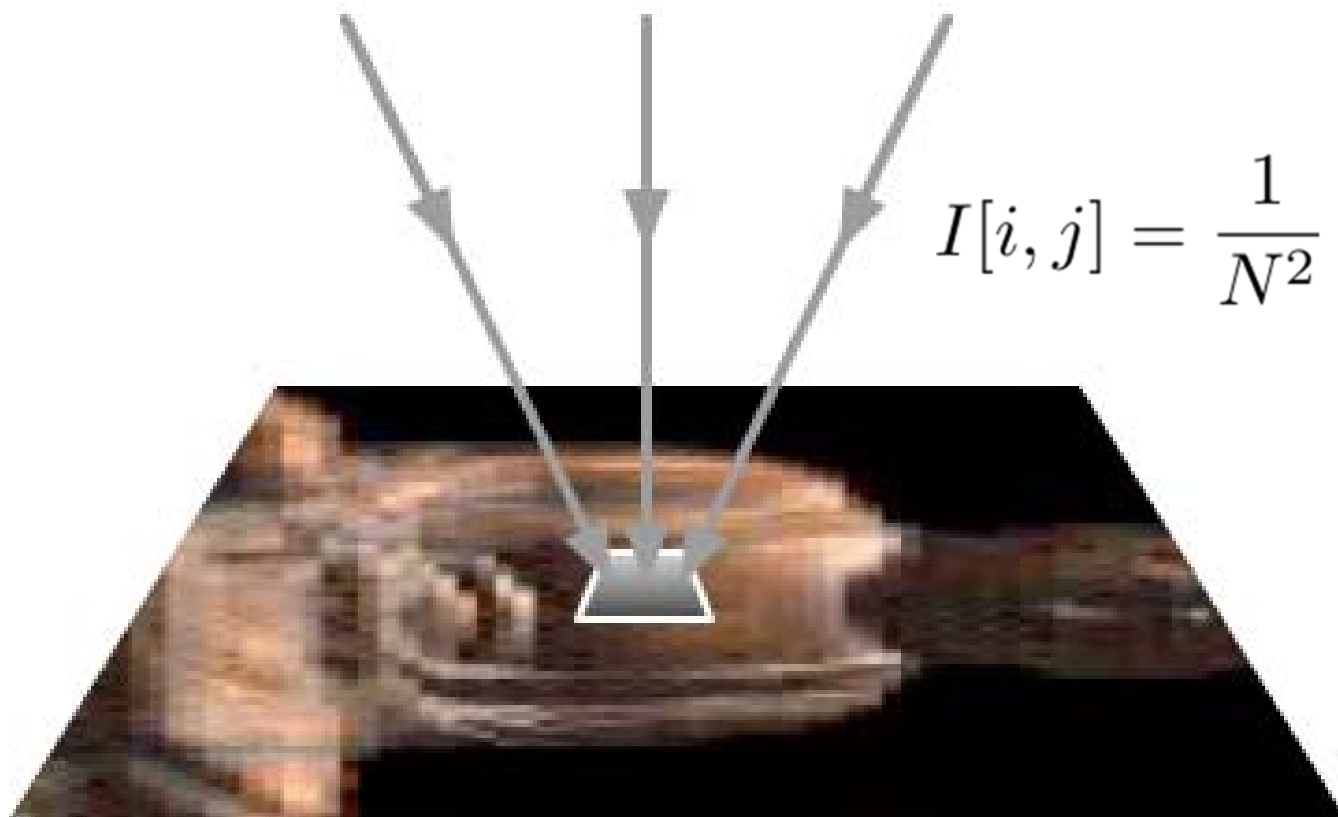


$$I(q) = \int_p r(q, p) dp$$



Image Rendering

- Integration is averaging over directions at each position



$$I[i, j] = \frac{1}{N^2} \sum_{m, n} r[m, n, i, j]$$

Image Rendering

- ▶ Integration is averaging over directions at each position

$$I[i, j] = \frac{1}{N^2} \sum_{m, n} r[m, n, i, j]$$

- ▶ Corresponding python code:

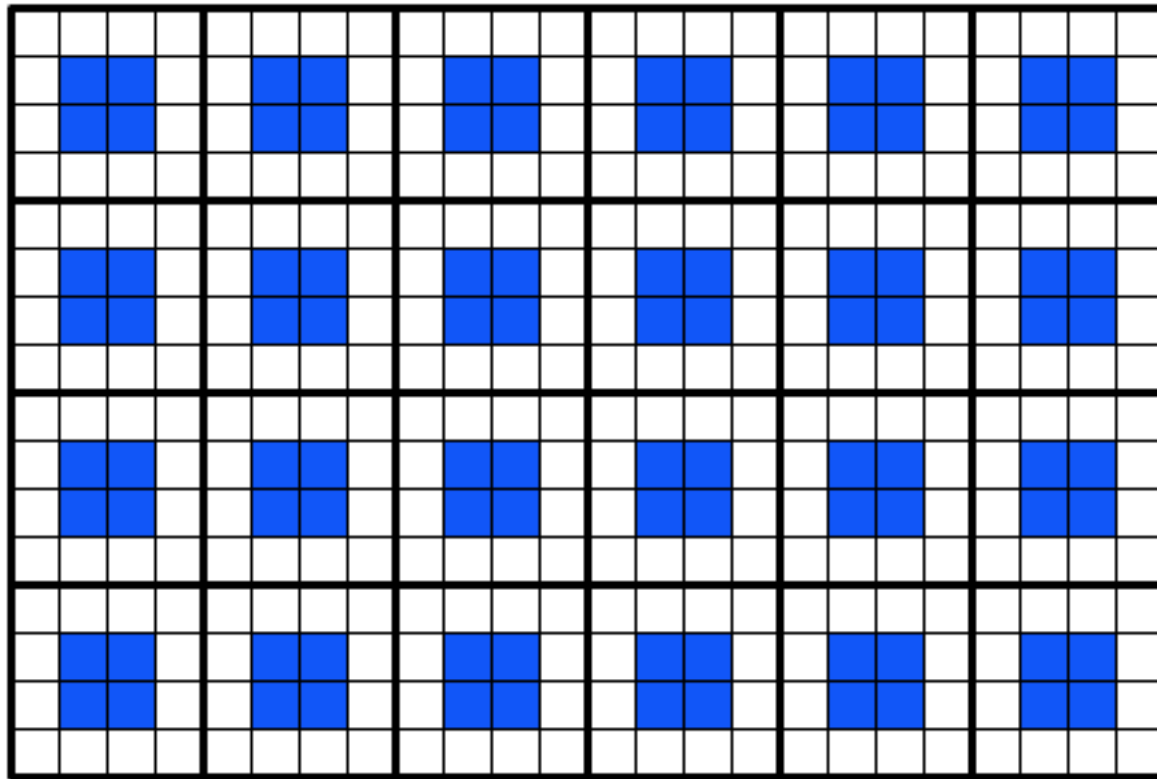
```
for j in range(0, nump):  
    for i in range(0, nump):  
        rendered[:, :] += radiance[:, :, j, i]  
  
rendered /= (nump * nump)
```


Rendering the Wide Variety

- ▶ Averaging recovers traditional picture
- ▶ Wide variety can also be rendered
 - ▶ Different aperture
 - ▶ Different viewpoint
 - ▶ Different focus
 - ▶ Different depth of field
 - ▶ Stereo
 - ▶ High dynamic range
 - ▶ Super resolution
 - ▶ ...

Different Aperture

- ▶ A smaller aperture is a smaller set of directions



Different Apertures

- ▶ A smaller aperture is a smaller set of directions

$$I[i, j] = \frac{1}{N^2} \sum_{m, n} r[m, n, i, j]$$

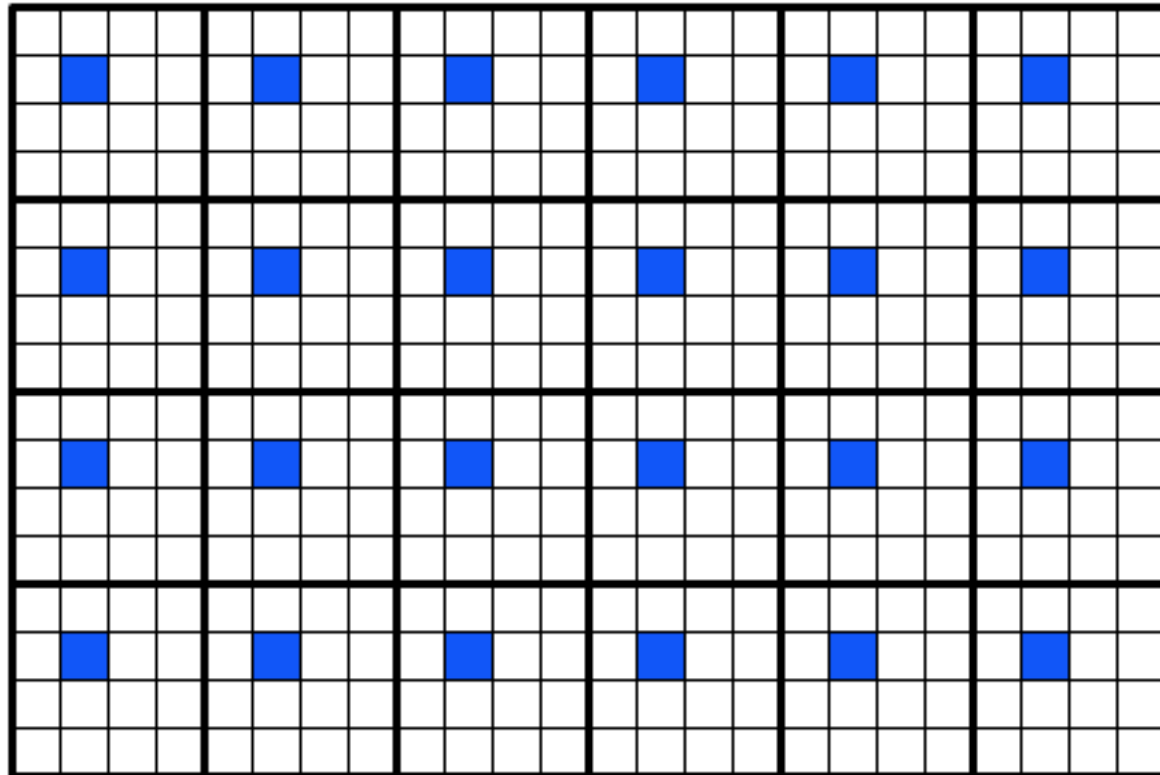
- ▶ Corresponding python code:

```
for j in range(alpha, nump-alpha):  
    for i in range(alpha, nump-alpha):  
        rendered[:, :] += radiance[:, :, j, i]  
  
rendered /= (nump * nump)
```

Pinhole Rendering (single viewpoint)

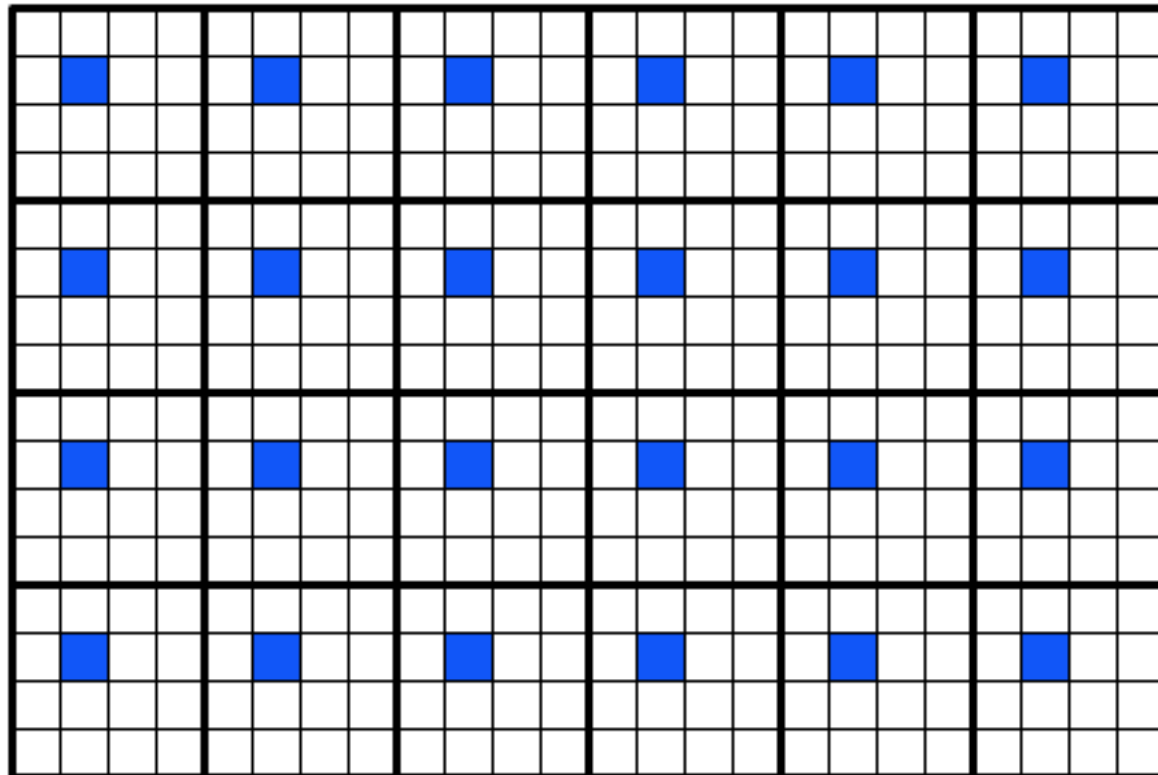
- ▶ Only render from one pixel from each microimage

```
rendered[:, :] = radiance[:, :, j, i]
```



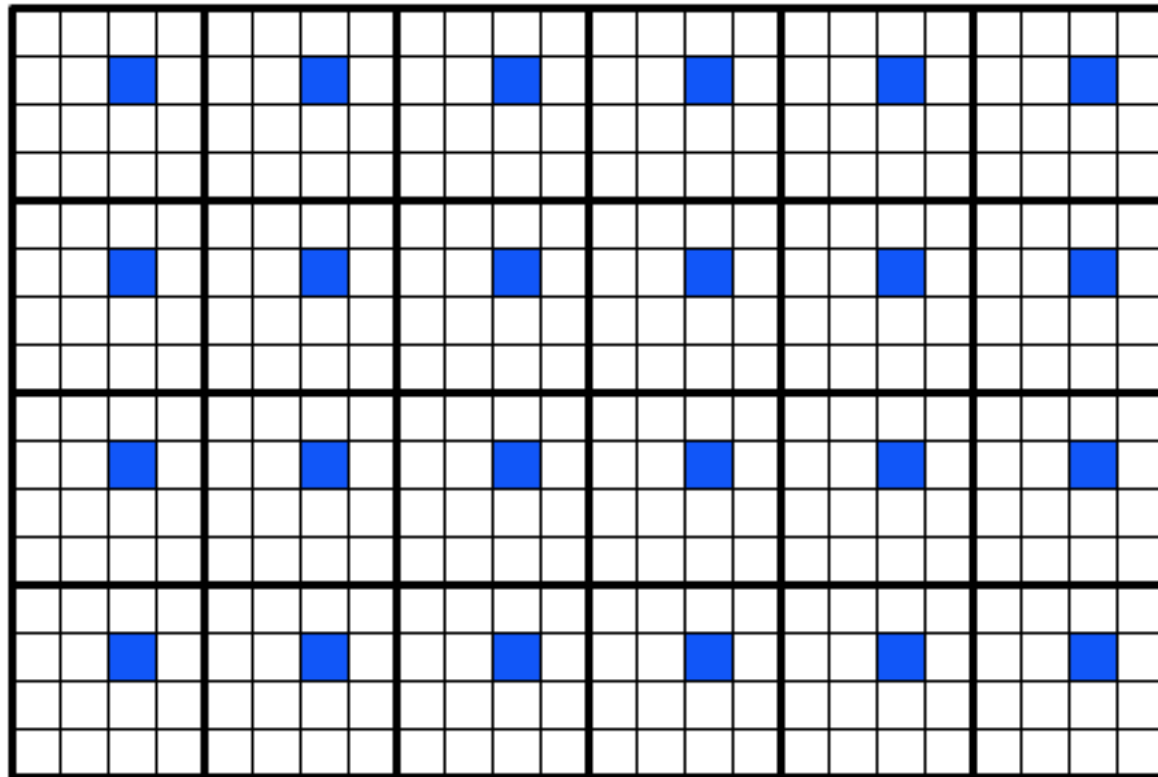
Different Viewpoints

- ▶ Different viewpoint is different direction
- ▶ Render different directions (or sets of directions)



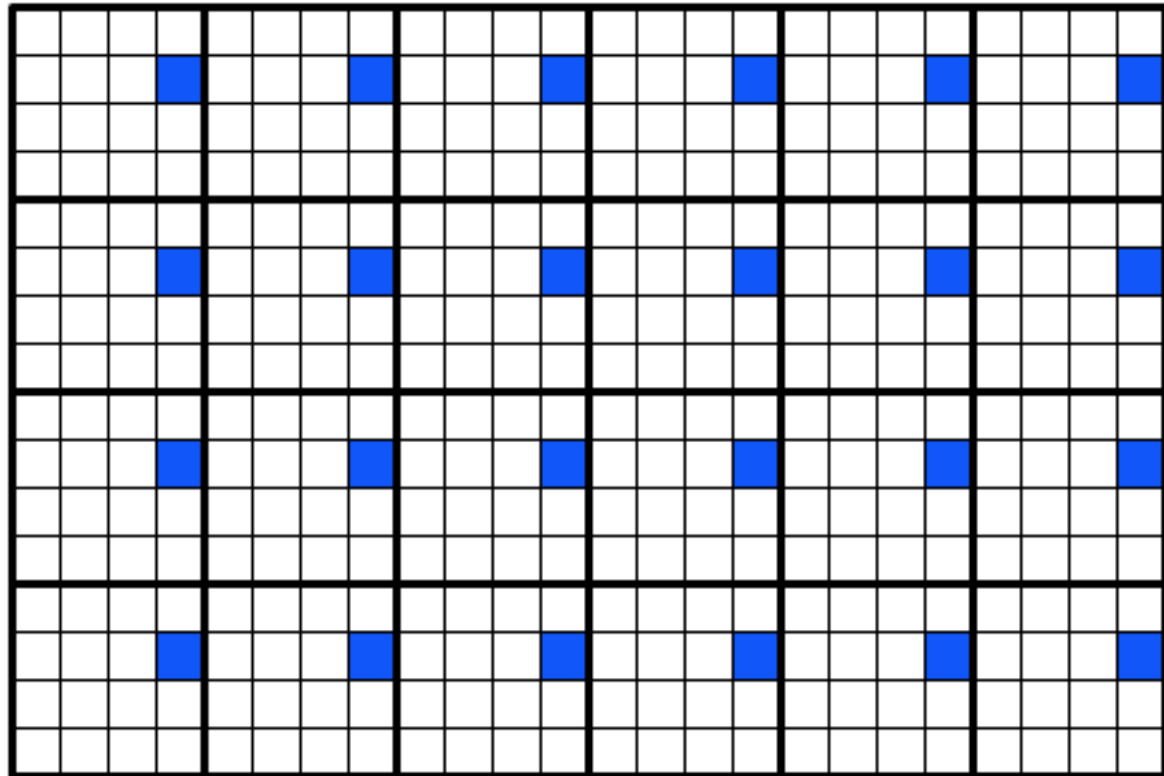
Different Viewpoints

- ▶ Different viewpoint is different direction
- ▶ Render different directions (or sets of directions)



Different Viewpoints

- ▶ Different viewpoint is different direction
- ▶ Render different directions (or sets of directions)





Example [Ren Ng]



Example [Ren Ng]

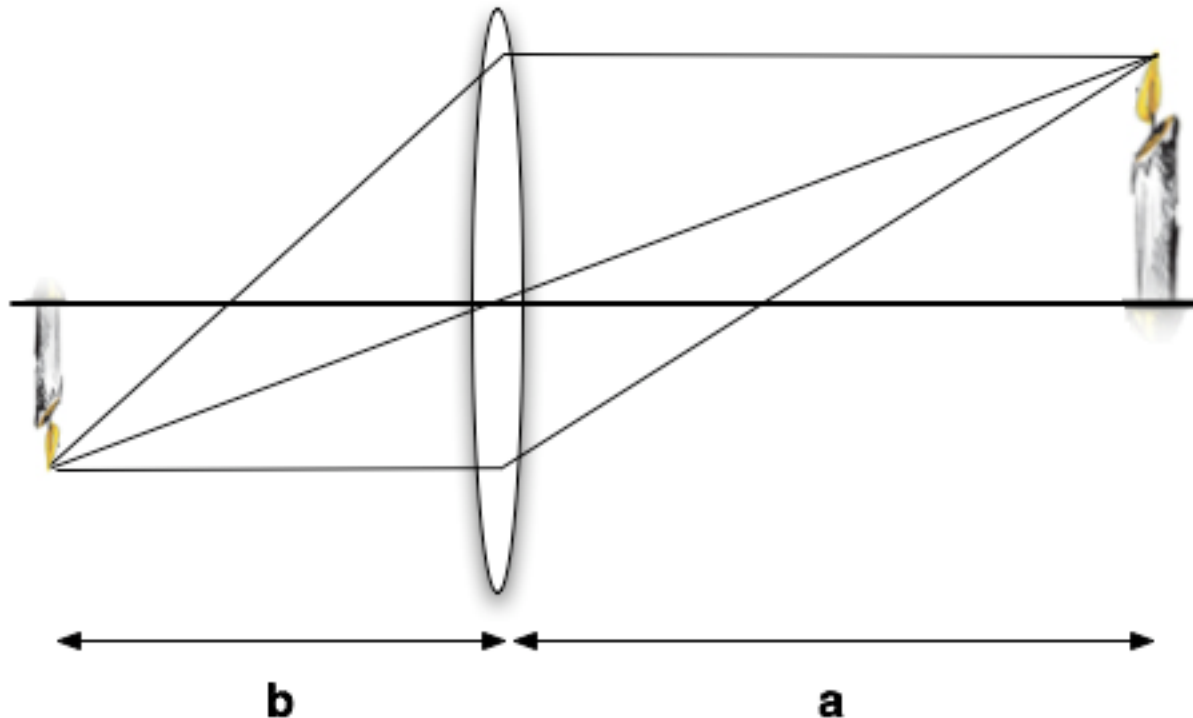


Example [Ren Ng]



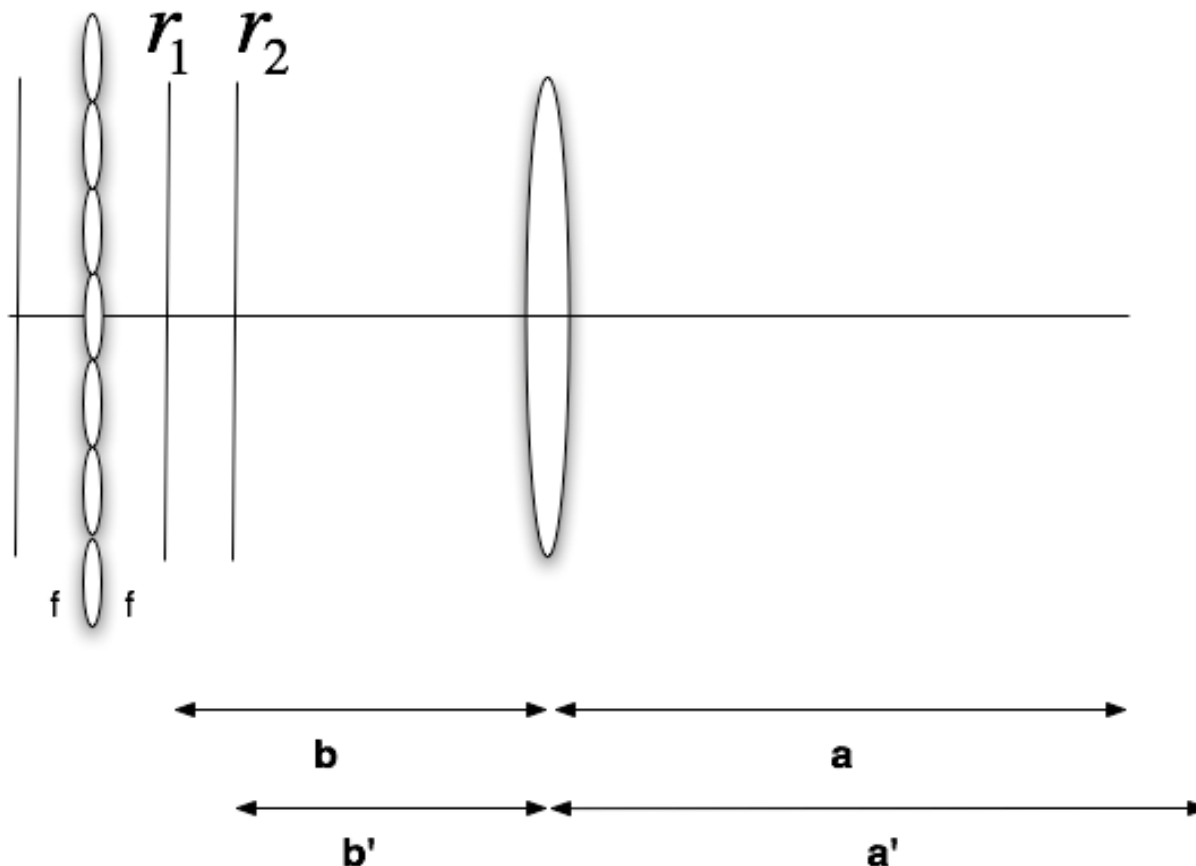
Refocusing

- ▶ When we refocus a camera, we change the distance from the lens to the sensor
- ▶ Same object is no longer in focus.



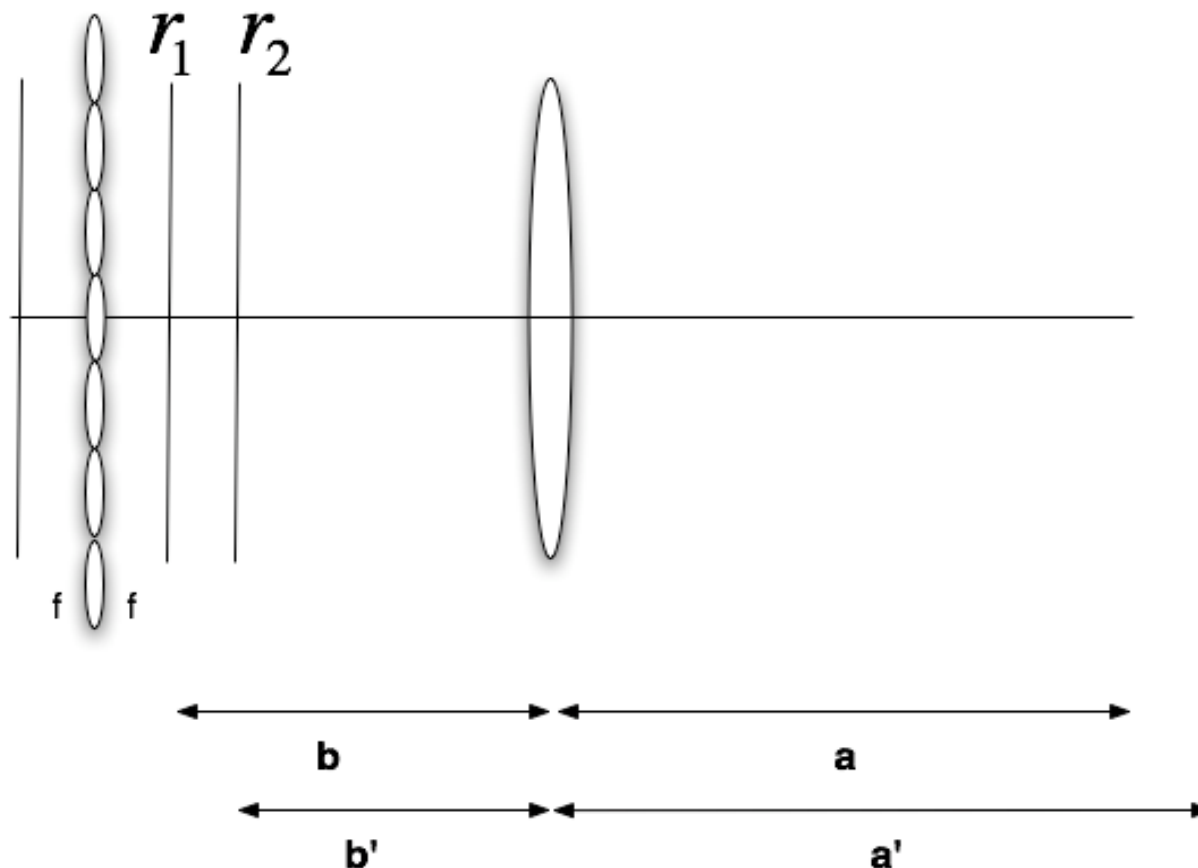
Computational Refocusing

- ▶ Change the distance (translate) computationally
- ▶ Two different radiances, r_1 and r_2



Computational Refocusing

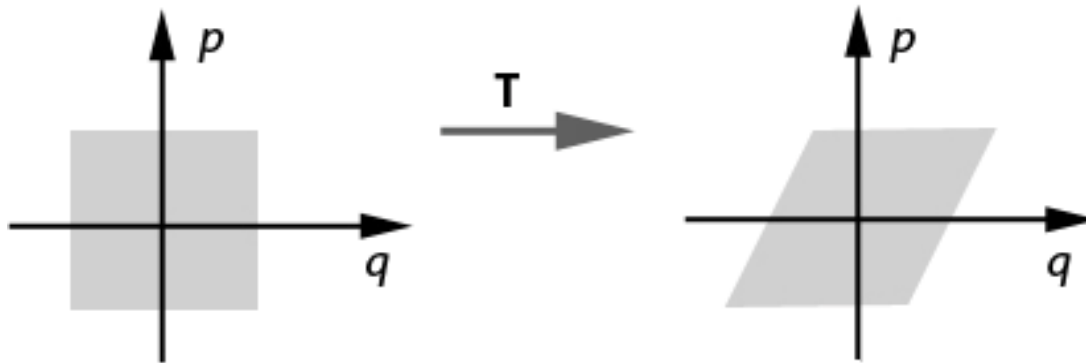
- ▶ We capture radiance r_1 . How can we compute r_2 ?
- ▶ We need translation transform of the radiance.





Algorithm: Computational Refocusing

- ▶ Apply shearing transformation: $r'(q, p) = r(q - tp, p)$



- ▶ Then render the new image: $I(q) = \int r(q, p) dp$

Algorithm: Refocusing

```
(yind,xind,wind,vind) = mgrid[0:m,0:n,0:r,0:s]
```

```
shear_y = y + t*wind / r
```

```
shear_x = x + t*vind / s
```

```
rad_p = interpolate(rad, [shear_y, shear_x, wind, vind])
```



Computational Refocusing (Ren Ng)



Computational Refocusing (Ren Ng)



Computational Refocusing (Ren Ng)



Fourier Slice Refocusing

Ng 2005

Efficient Computational Refocusing

- ▶ Refocusing in the spatial domain requires $O(N^4)$ operations for each refocused image
- ▶ An alternative approach (invented by Ren Ng) requires $O(N^4 \log(N))$ for initial setup but then for each rendered image we need only $O(N^2 \log(N))$
- ▶ Insight: Refocus in the frequency domain
- ▶ The frequency domain representation of the rendering integral is the DC directional component (slice)

Transform of Rendered Image

- ▶ The Fourier transform of a rendered image:

$$\hat{I}(\omega_q) = \int I(q) e^{i\omega_q \cdot q} dq$$

- ▶ Recall that $I(q) = \int_p r(q, p) dp$

- ▶ Thus we have
$$\begin{aligned} \hat{I}(\omega_q) &= \int \int r(q, p) e^{i\omega_q \cdot q} dp dq \\ &= R(\omega_q, 0) \end{aligned}$$

- ▶ In other words, the transform of the rendered image is the DC directional component of $R(\omega)$.

Translation in the Frequency Domain

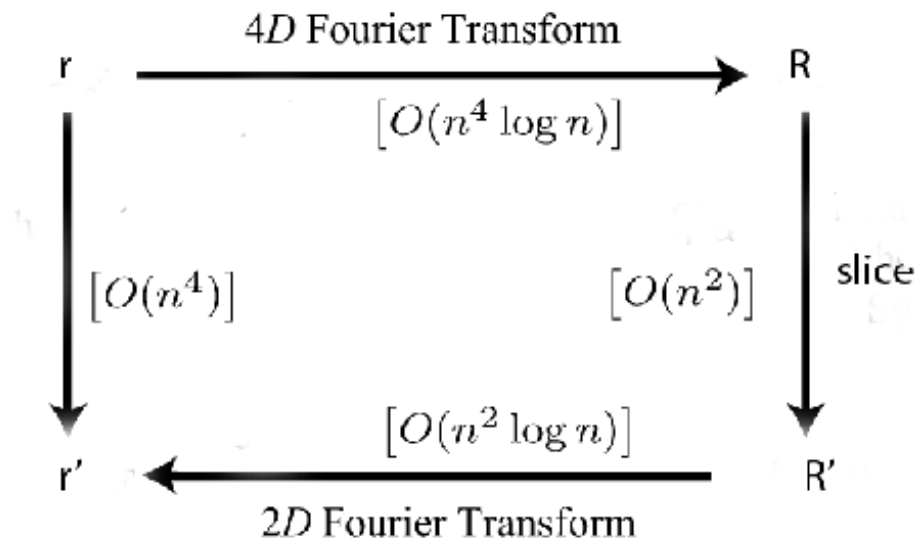
- ▶ Recall $R'(\omega) = R(A^T \omega)$
- ▶ In the case of translation

$$R'(\omega_q, \omega_p) = R(\omega_q, \omega_p - t\omega_q)$$

- ▶ But we are interested in the case $\omega_p = 0$
- ▶ I.e., $I'(\omega_q) = R'(\omega_q, 0) = R(\omega_q, -t\omega_q)$
- ▶ The refocused image is just a **slice** (with slope t)

Algorithm: Fourier Slice Refocusing

- ▶ Take FFT of radiance: $R[i, j, m, n] = FFT(r[i, j, m, n])$
- ▶ Interpolate to get a slice: $R[i, j, m, n] \rightarrow \hat{I}[i, j]$
- ▶ Take inverse FFT: $I'[i, j] = IFFT(\hat{I}[i, j])$



Algorithm: Fourier Slice Refocusing

```
radiancefft = fftn(radiance)
```

```
(yind,xind) = mgrid[0:m,0:n]
```

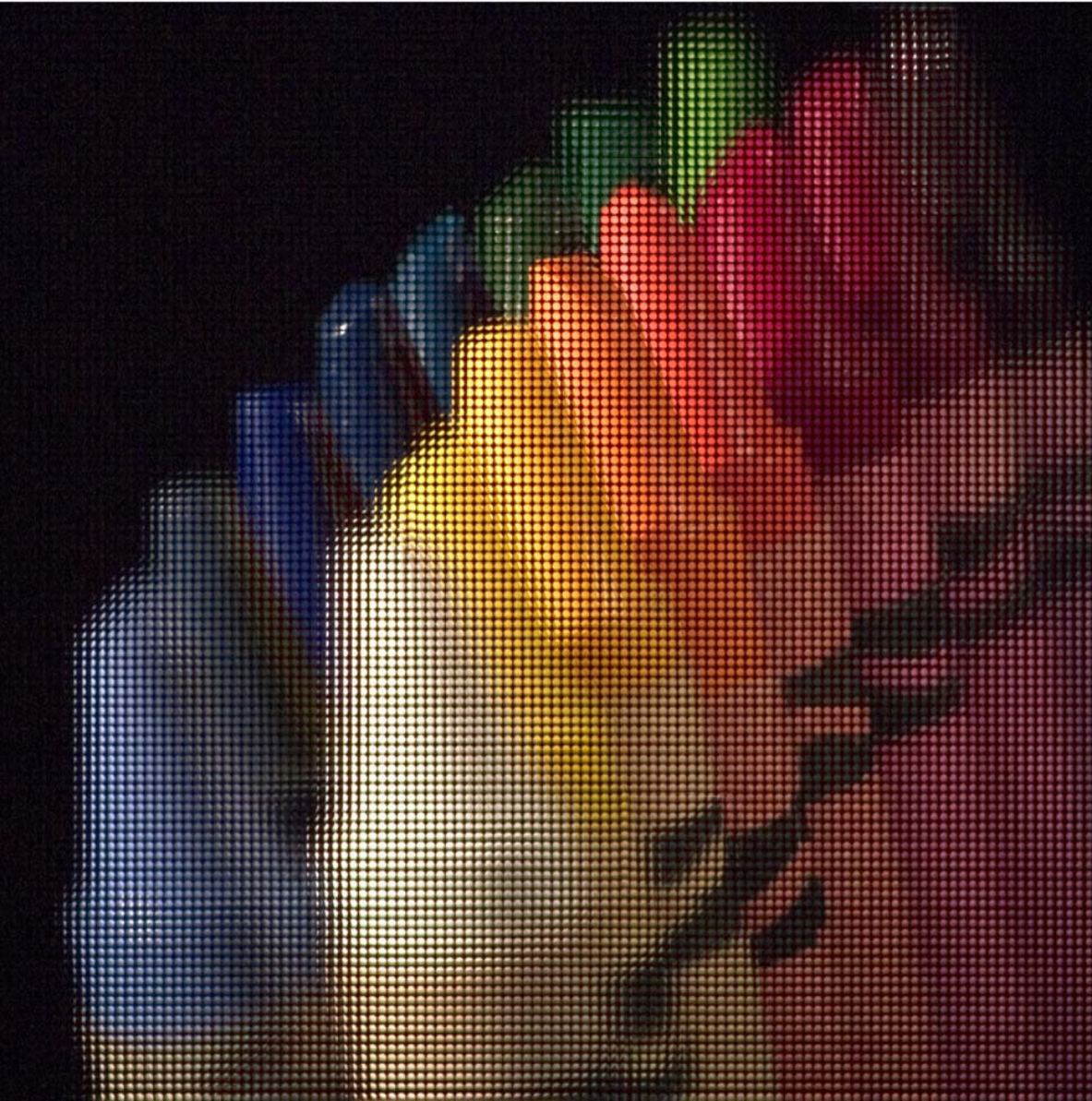
```
vind = t*yind / m
```

```
uind = t*xind / n
```

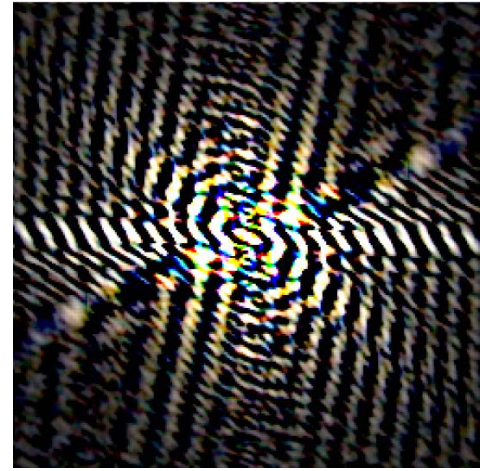
```
slice = interpolate(radiancefft, [yind, xind, vind, uind])
```

```
rendered = ifft2(slice)
```

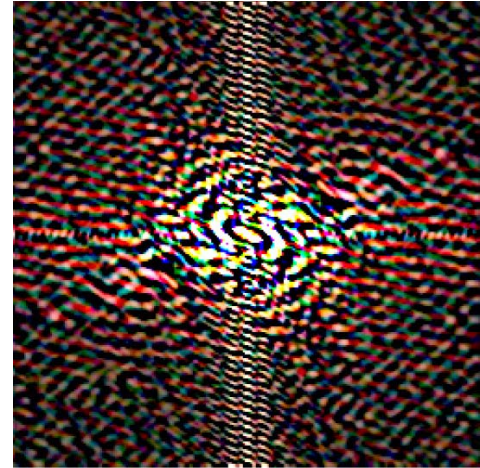

Fourier Slice Refocusing (Ren Ng)



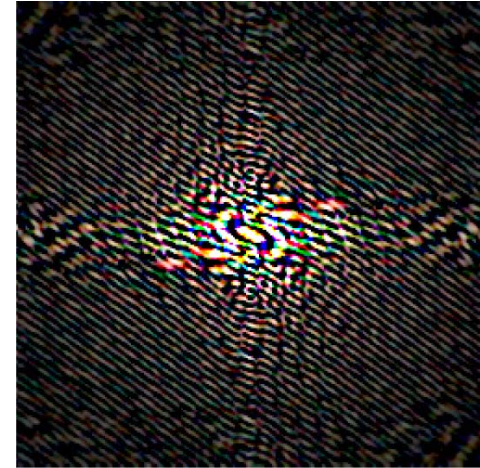
Fourier Slice Refocusing (Ren Ng)



Fourier Slice Refocusing (Ren Ng)



Fourier Slice Refocusing (Ren Ng)



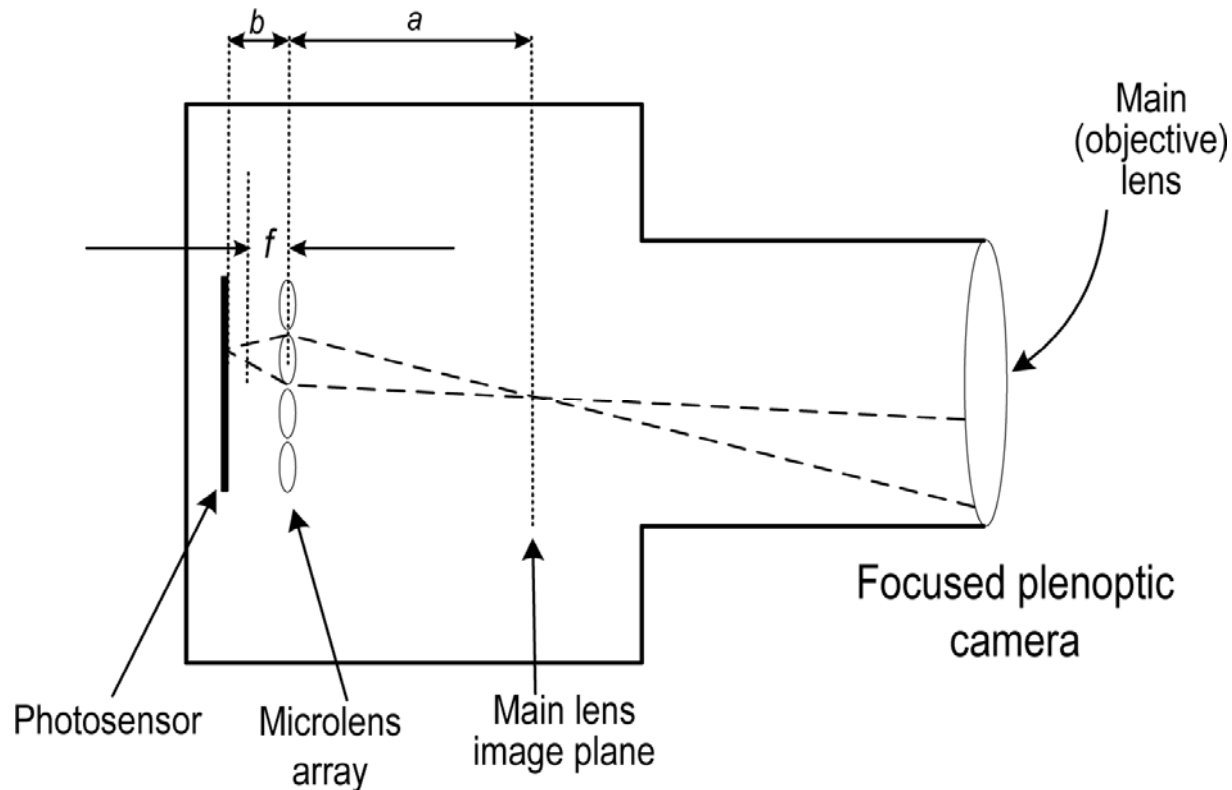


Methods for Plenoptic 2.0



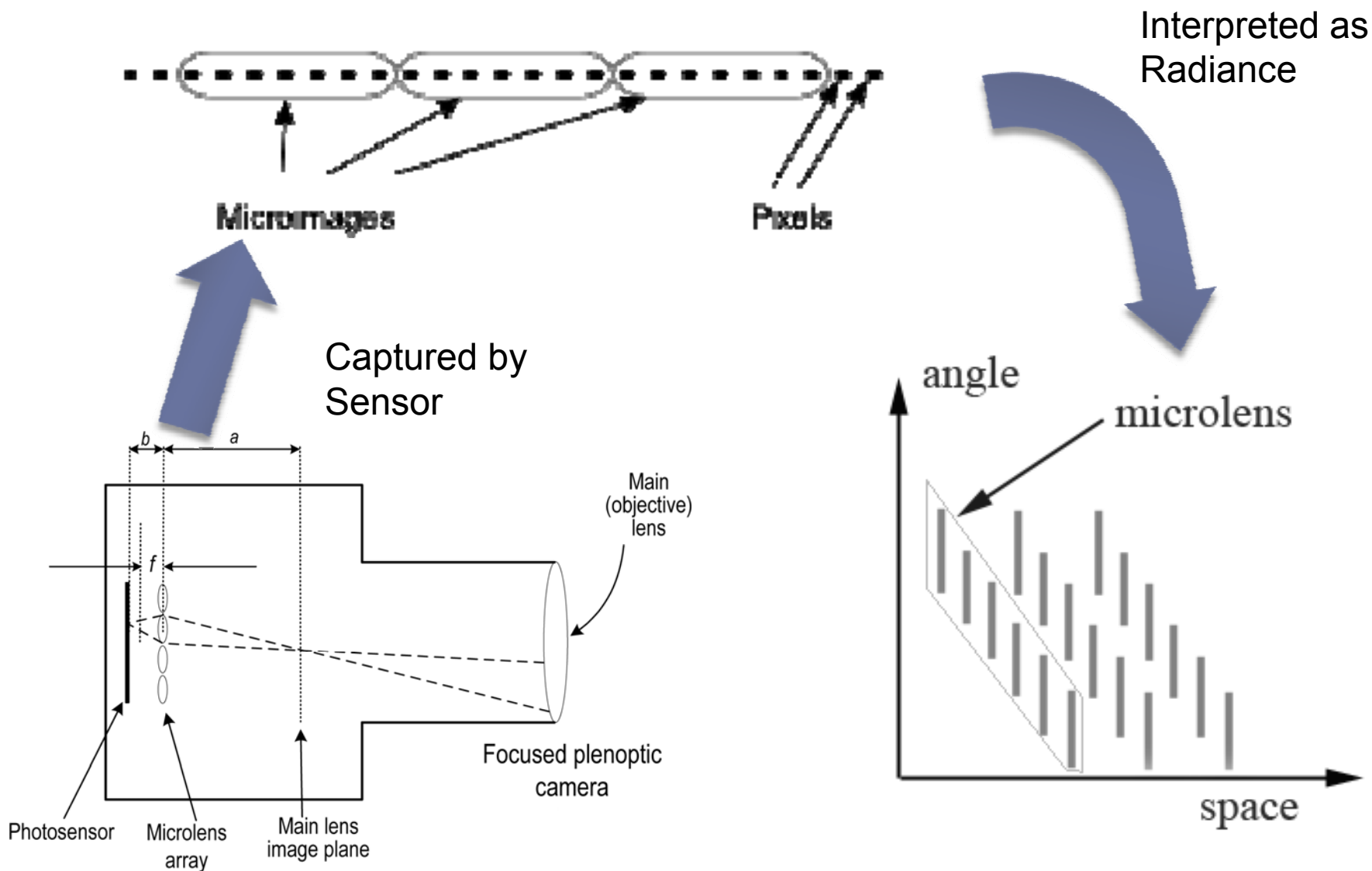
Radiance Representation (Plenoptic 2.0)

- ▶ Sensor image samples the radiance
 - ▶ Each microlens image samples in position and direction



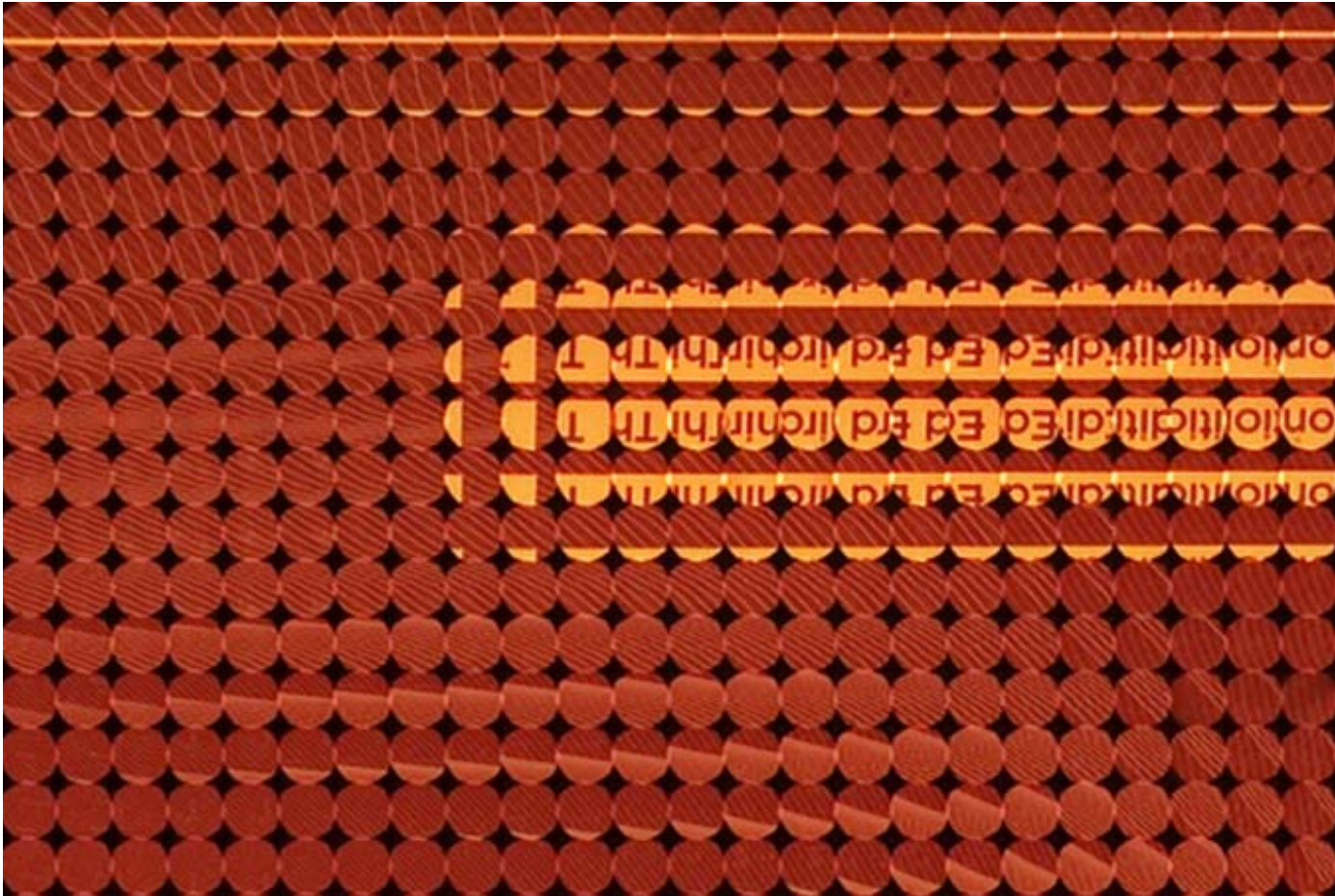


Radiance Representation (Plenoptic 2.0)





Radiance Representation (Plenoptic 2.0)

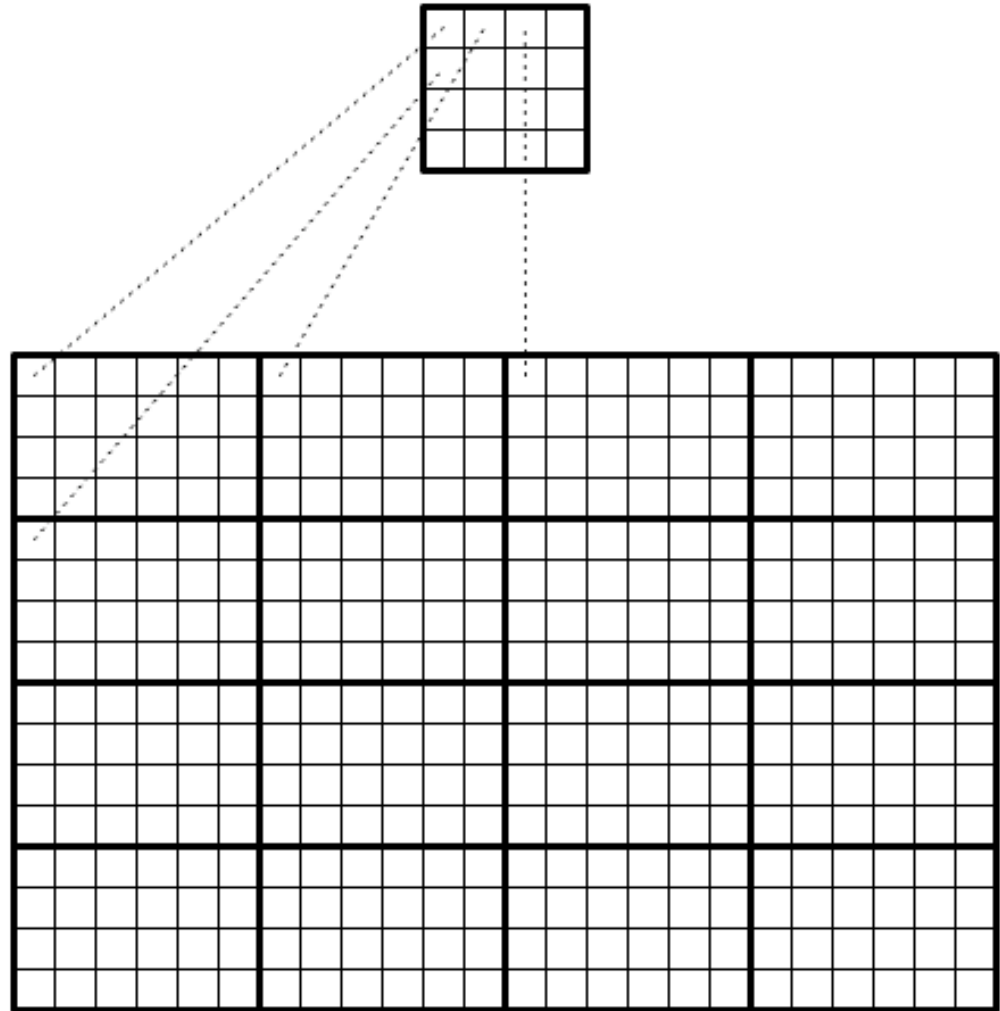


Radiance Representation (Plenoptic 2.0)

- ▶ Plenoptic 2.0 image is a “flat” 2D array of 2D arrays
- ▶ Radiance is 4D array
- ▶ “Direction major” (approximately, in the sense of tilting)

Creating Radiance Array (Direction Major)

- ▶ Given 2D “flat” radiance captured by plenoptic camera
- ▶ Create 4D array
- ▶ Sample the same positional pixel from every direction
- ▶ Convention $r[i, j, m, n]$
 - ▶ Follow $r(q, p)$
 - ▶ i, j are position
 - ▶ m, n are direction



Creating Radiance Array (Direction Major)

- ▶ Given 2D direction major “flat” from radiance camera
- ▶ Create 4D array
- ▶ If 2D direction major “flat” is regular

```
(jnds,inds) = mgrid[0:height:numq,0:width:numq]
```

```
for j in range(0,numq):
```

```
    for i in range(0,numq):
```

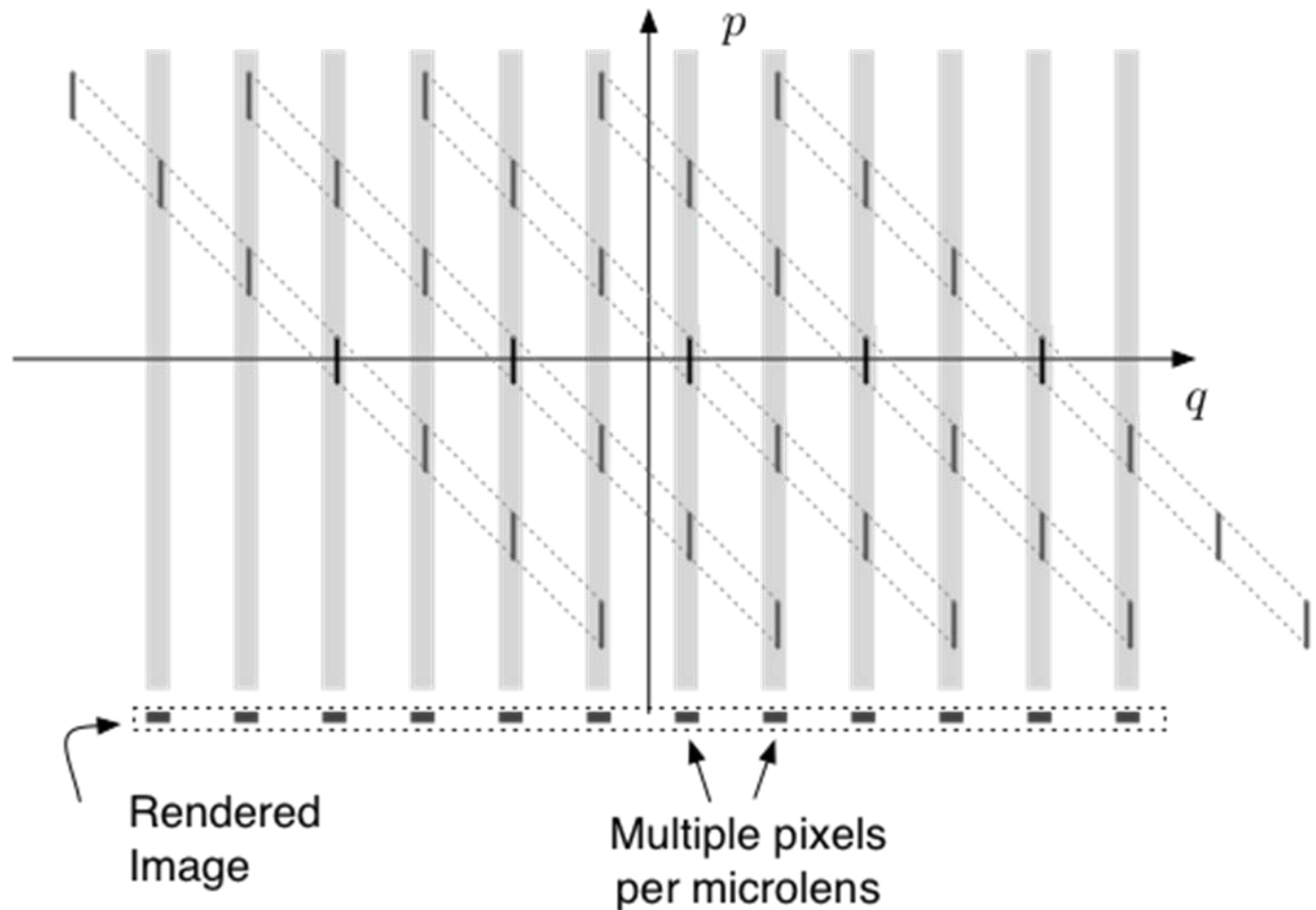
```
        radiance[j,l,:,:) = image[jnds+j,inds+i]
```

- ▶ Python, matlab very similar
- ▶ Samples the same positional pixel from every direction



Rendering Full Aperture Image from 2.0 Data

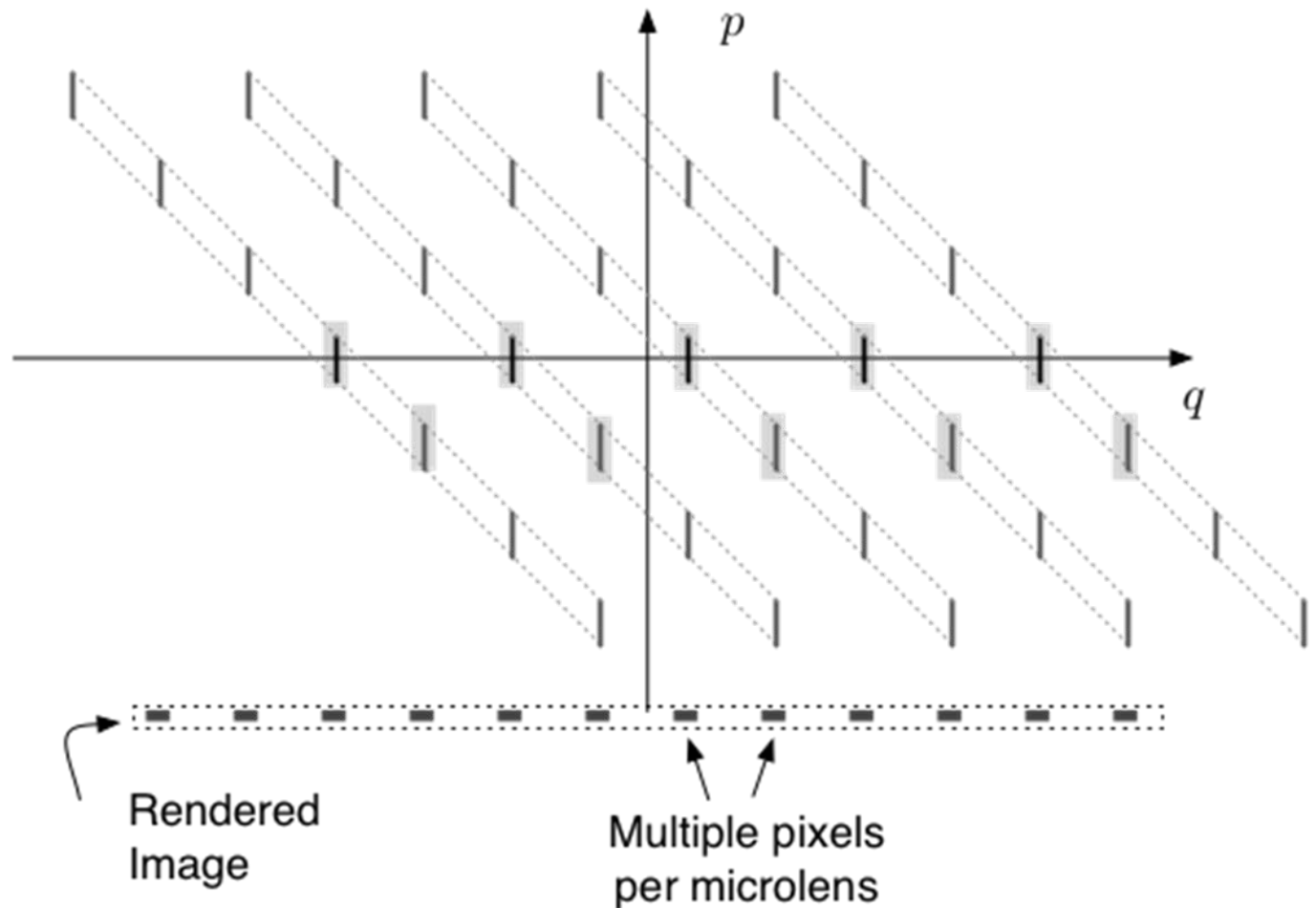
$$I[i, j] = \frac{1}{N^2} \sum_{m, n} r[m, n, i, j]$$



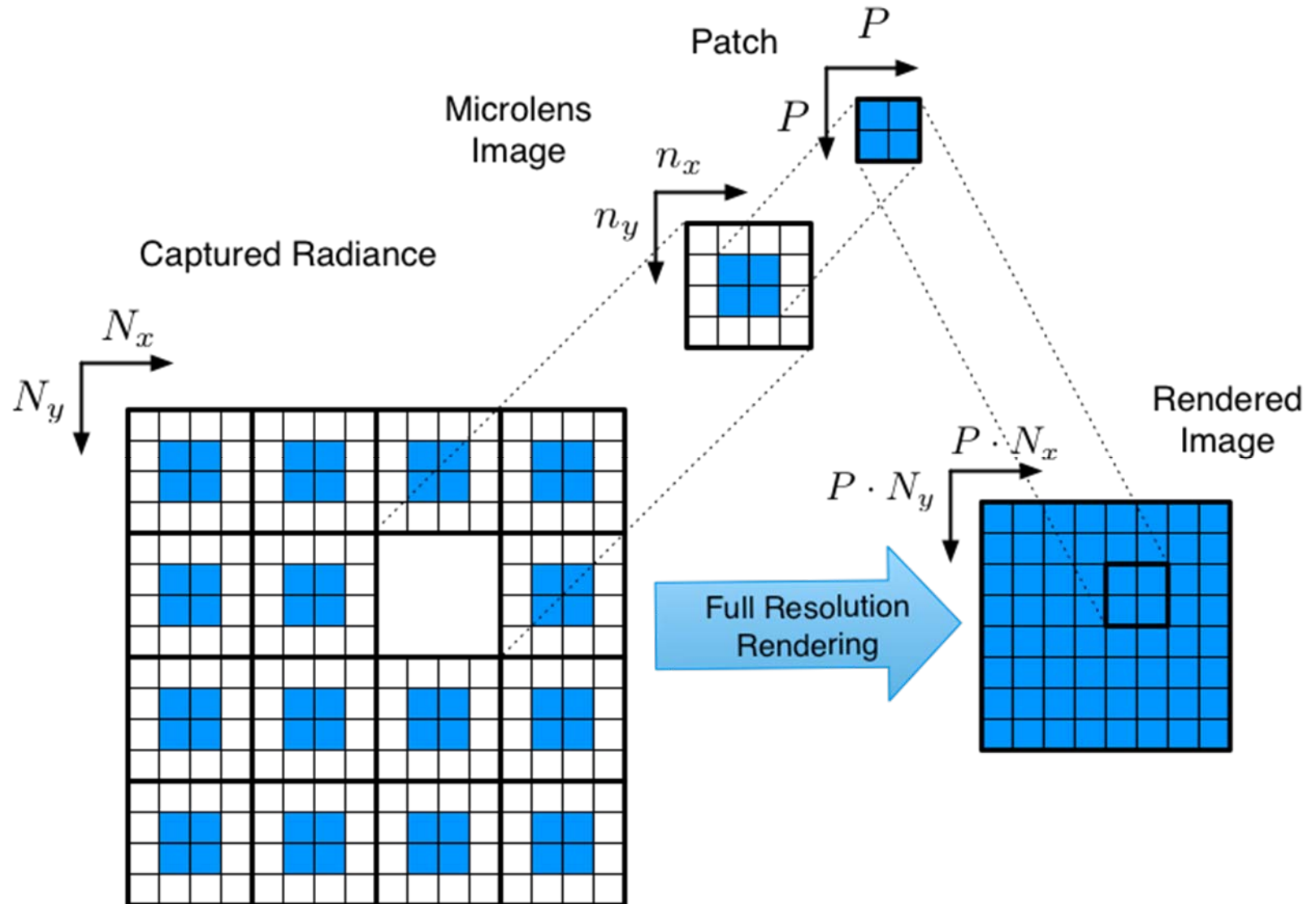


Rendering One View from 2.0 Data

$$I[i, j] = r[m, n, i, j]$$

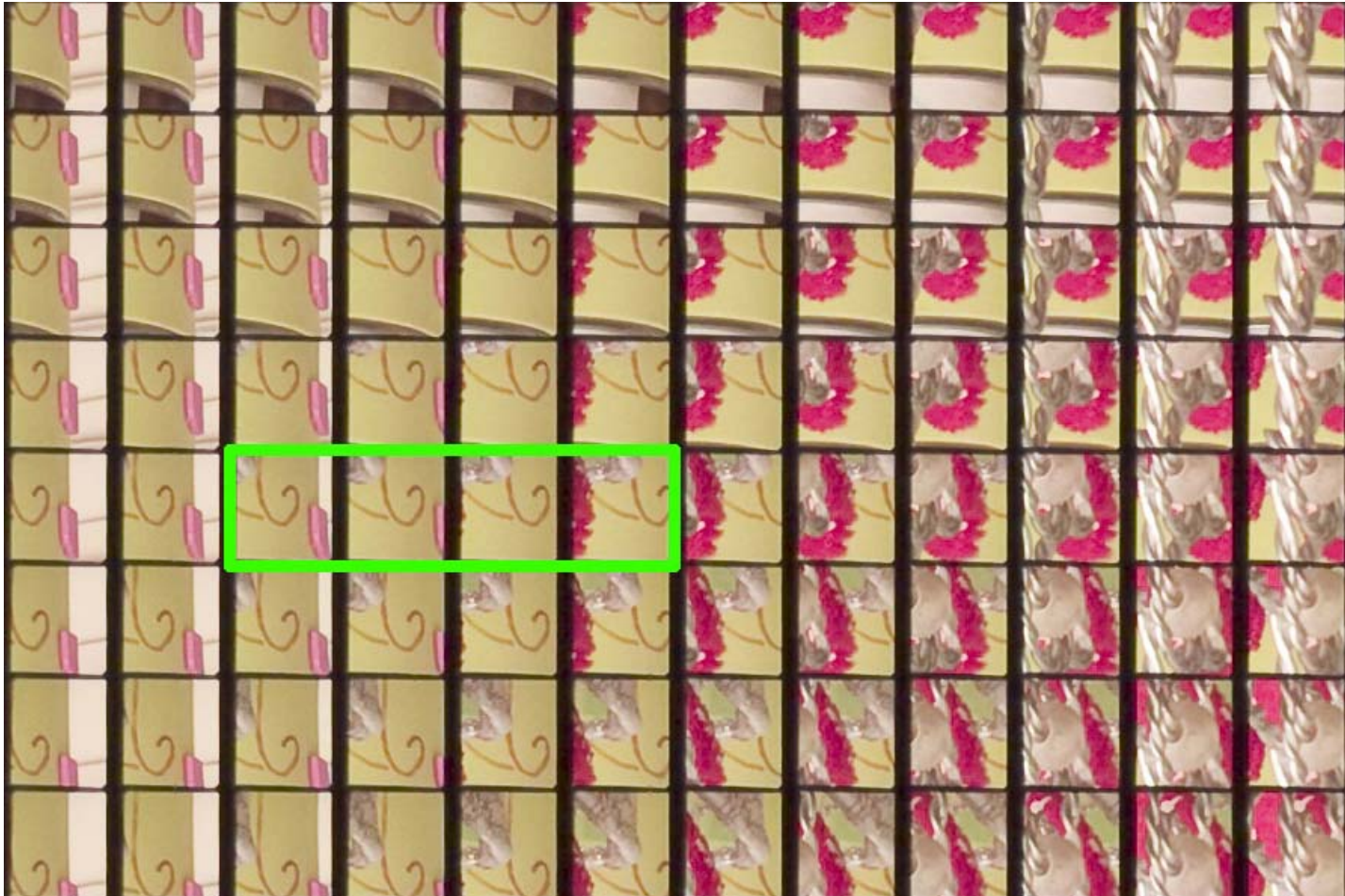


Plenoptic 2.0 Rendering



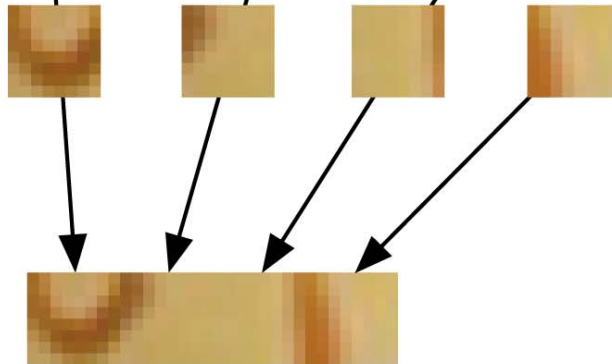
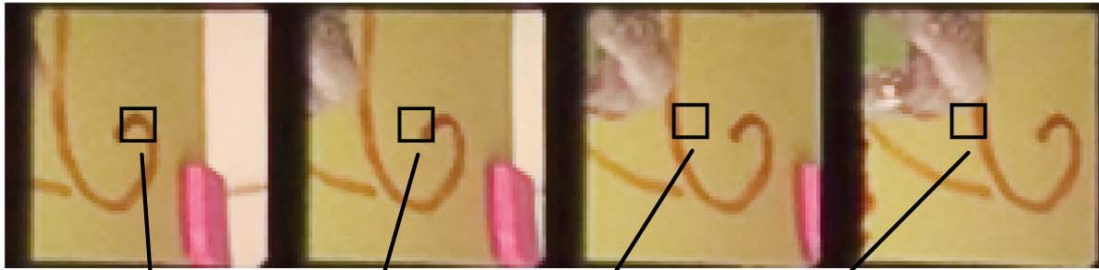


Plenoptic 2.0 Rendering Example





Plenoptic 2.0 Rendering Example

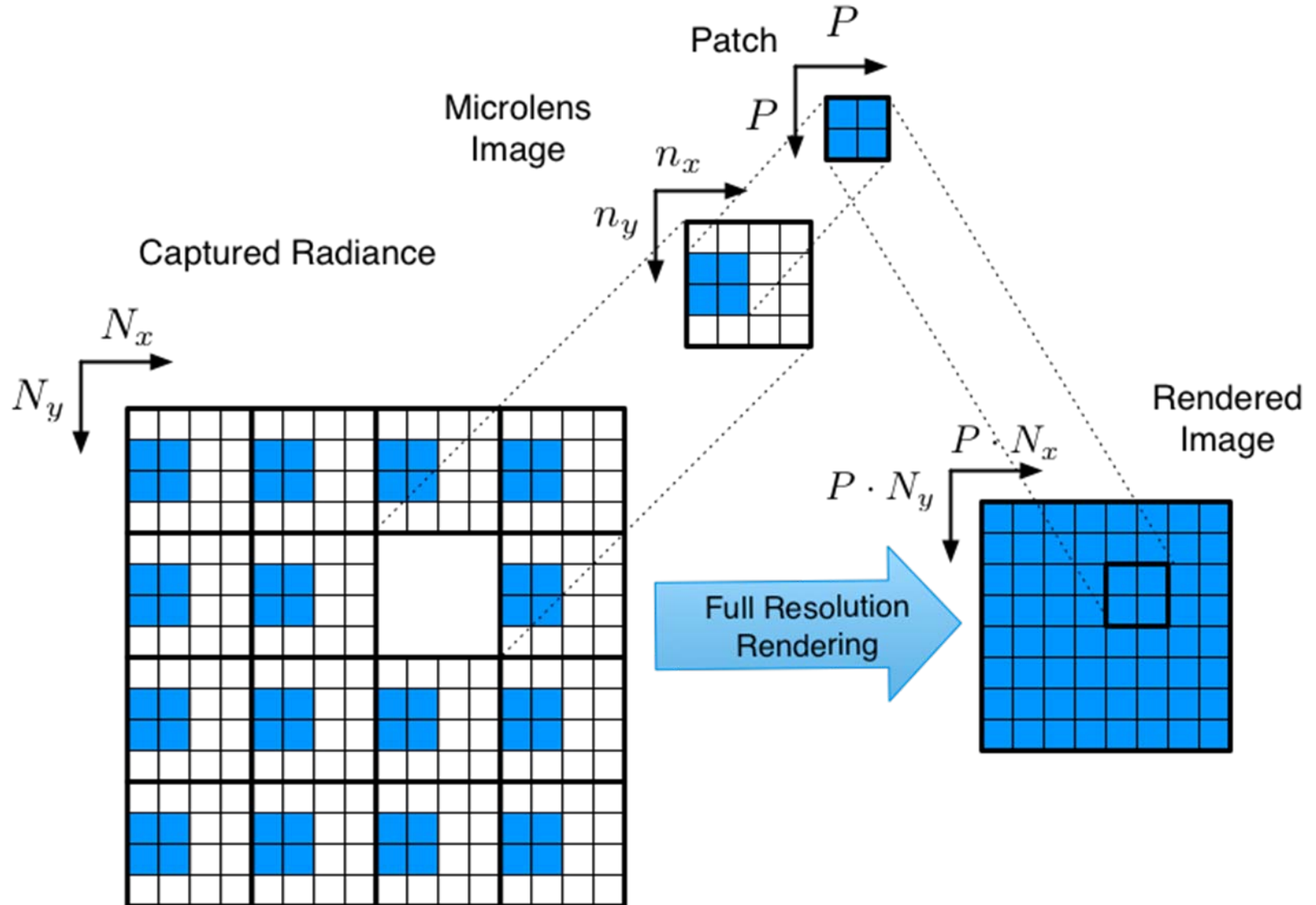




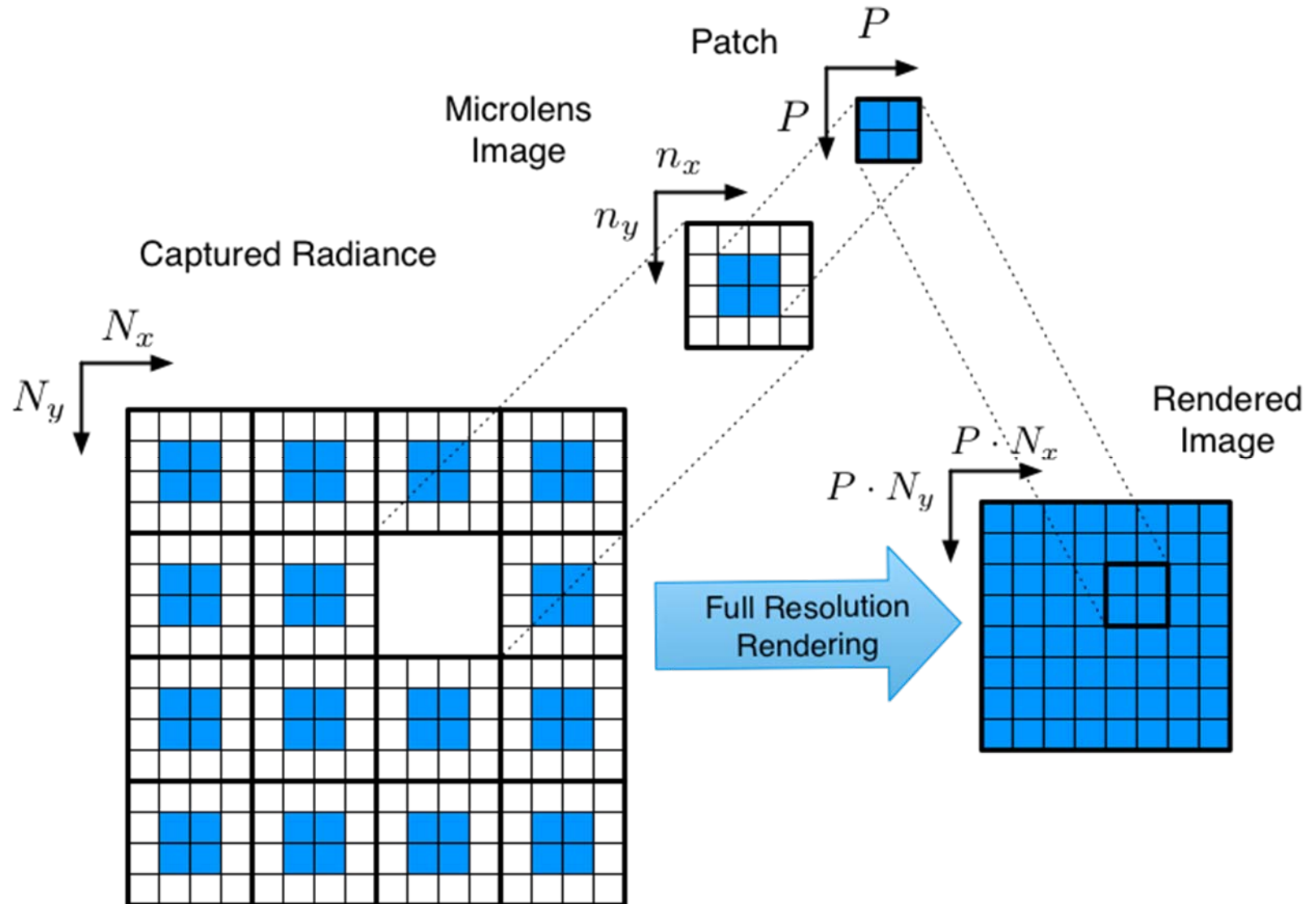
Rendering the Wide Variety

- ▶ Averaging recovers traditional picture
- ▶ Wide variety can also be rendered
 - ▶ Different aperture
 - ▶ Different viewpoint
 - ▶ Different focus
 - ▶ Different depth of field
 - ▶ Stereo
 - ▶ High dynamic range
 - ▶ Super resolution
 - ▶ ...

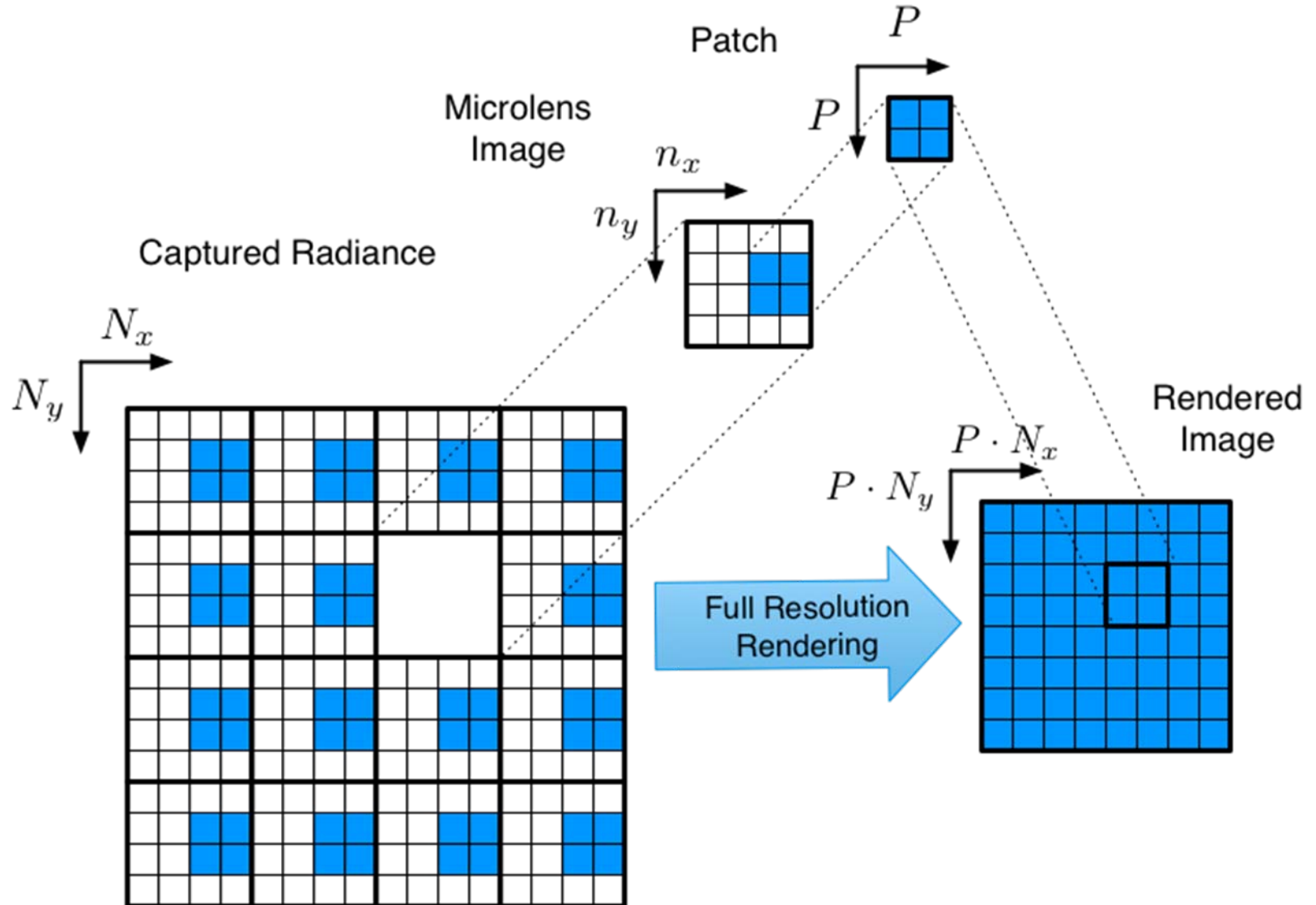
Plenoptic 2.0 Rendering Parallax



Plenoptic 2.0 Rendering Parallax

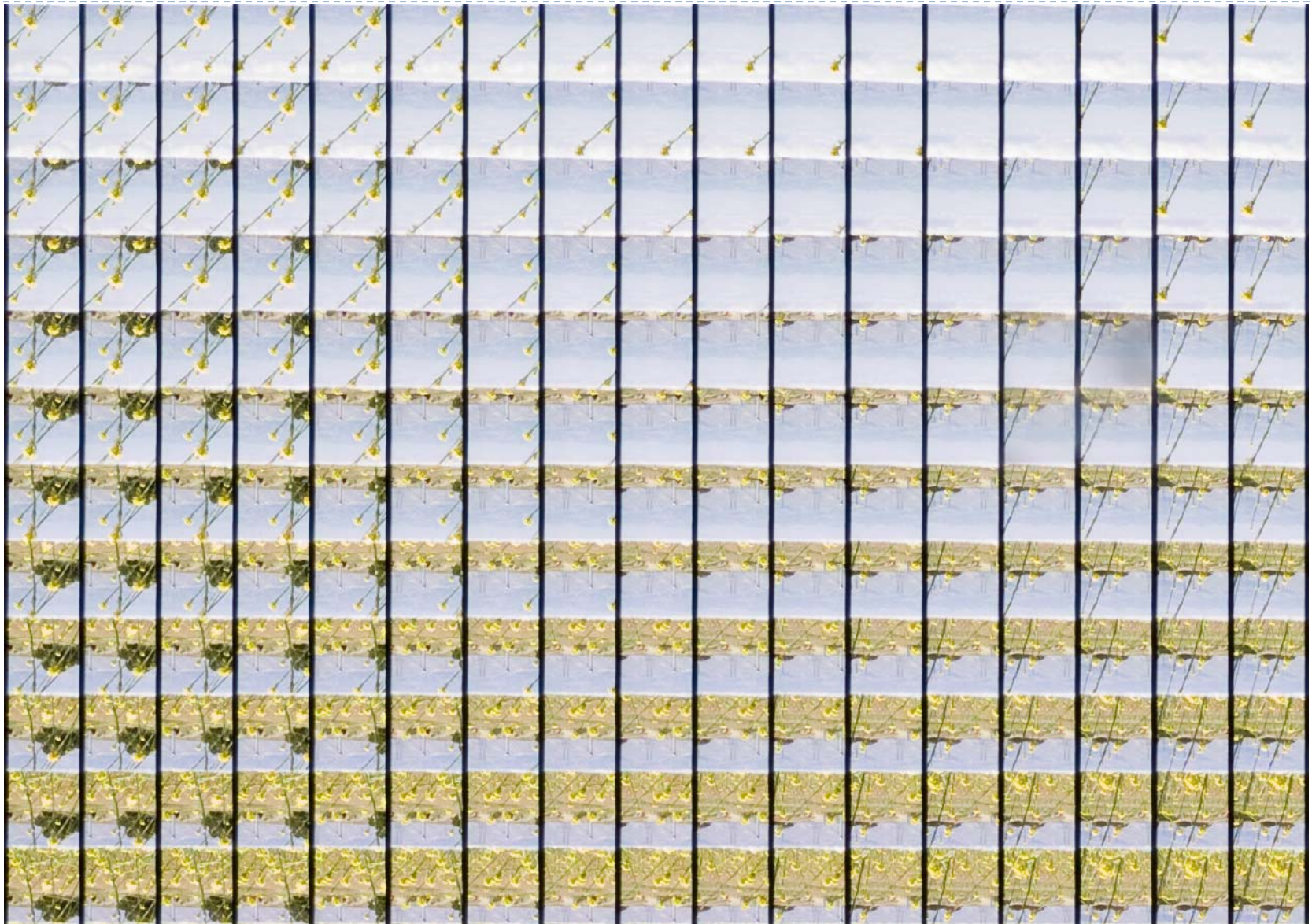


Plenoptic 2.0 Rendering Parallax





Plenoptic 2.0 Rendering Parallax





Plenoptic 2.0 Rendering Parallax





Plenoptic 2.0 Rendering Parallax





Plenoptic 2.0 Refocusing





Plenoptic 2.0 Refocusing





Plenoptic 2.0 Refocusing





Plenoptic 2.0 Refocusing

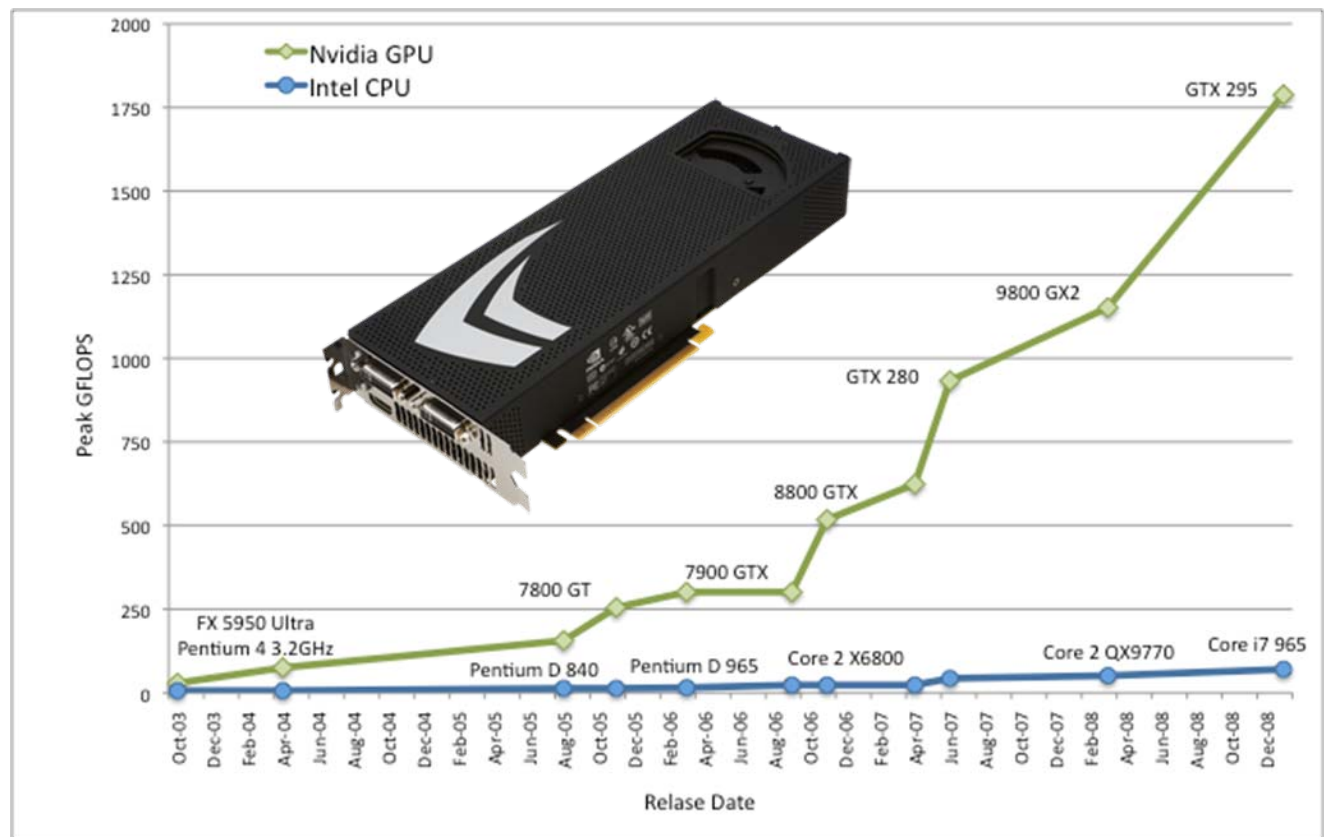


Efficient Implementation with GPU

Real-Time Radiance Rendering and Transforms

Graphics Processing Units

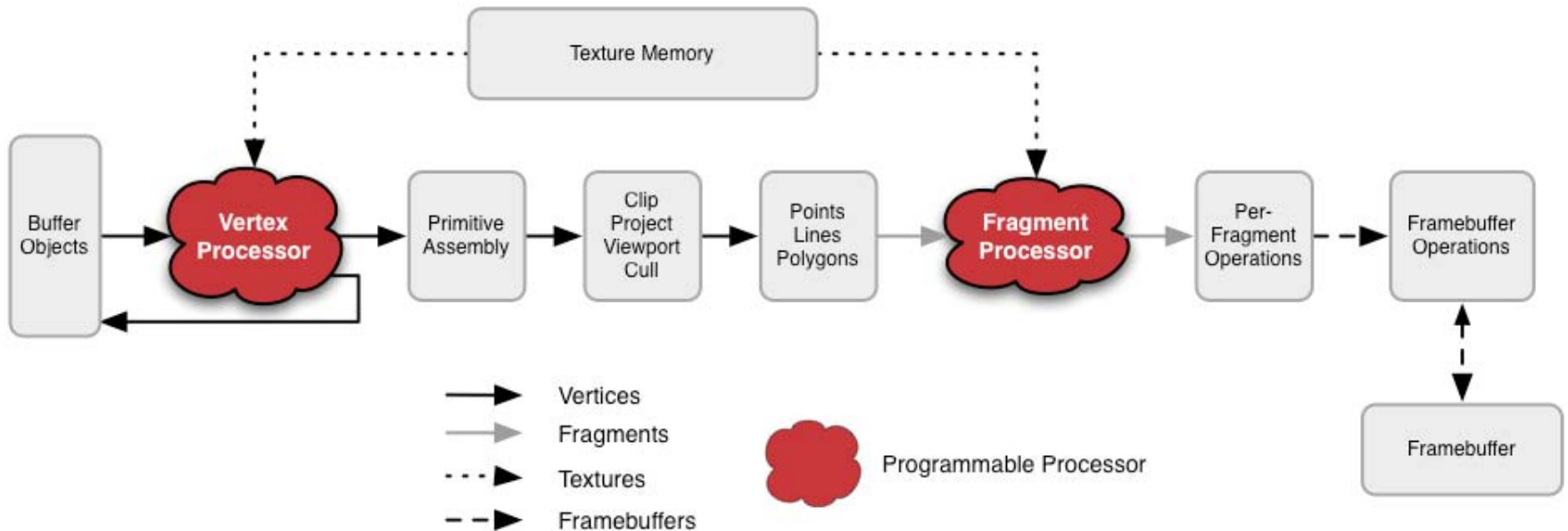
- ▶ Radiance processing is computationally expensive
- ▶ CPU clock speeds stalled at 3.0GHz
- ▶ Nvidia GTX 295:
 - ▶ 1.8 Tflop
 - ▶ \$500





GPU Programming

- ▶ Basic alternatives for programming GPU: General purpose or graphics-based
- ▶ Open GL Shader Language (GLSL) a natural fit





Rendering with GPU using Open GL

- ▶ Read in plenoptic radiance image
- ▶ Create 2D texture object for radiance
- ▶ Serialize image data to Open GL compatible format
- ▶ Define the texture to OpenGL

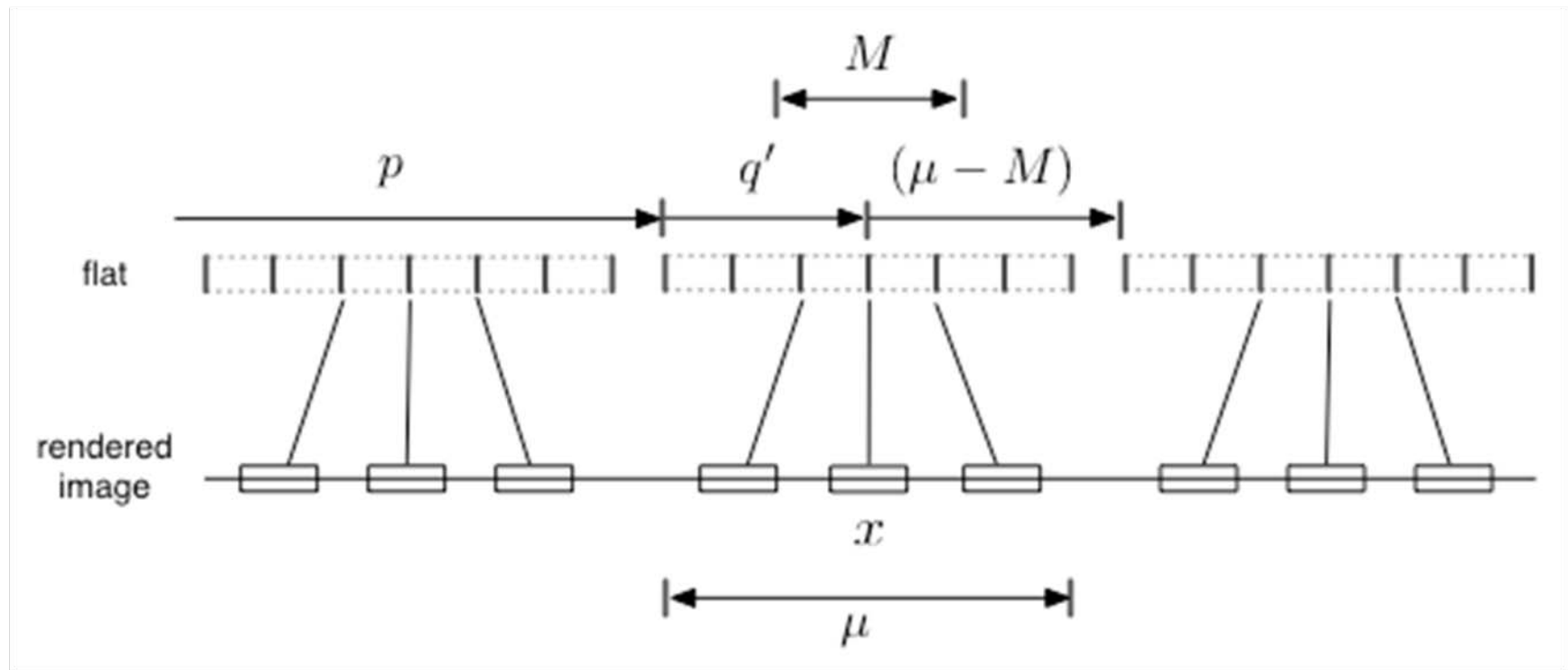
```
image = Image.open("lightfield.png")

str_image = image.tostring("raw", "RGBX", 0, -1)

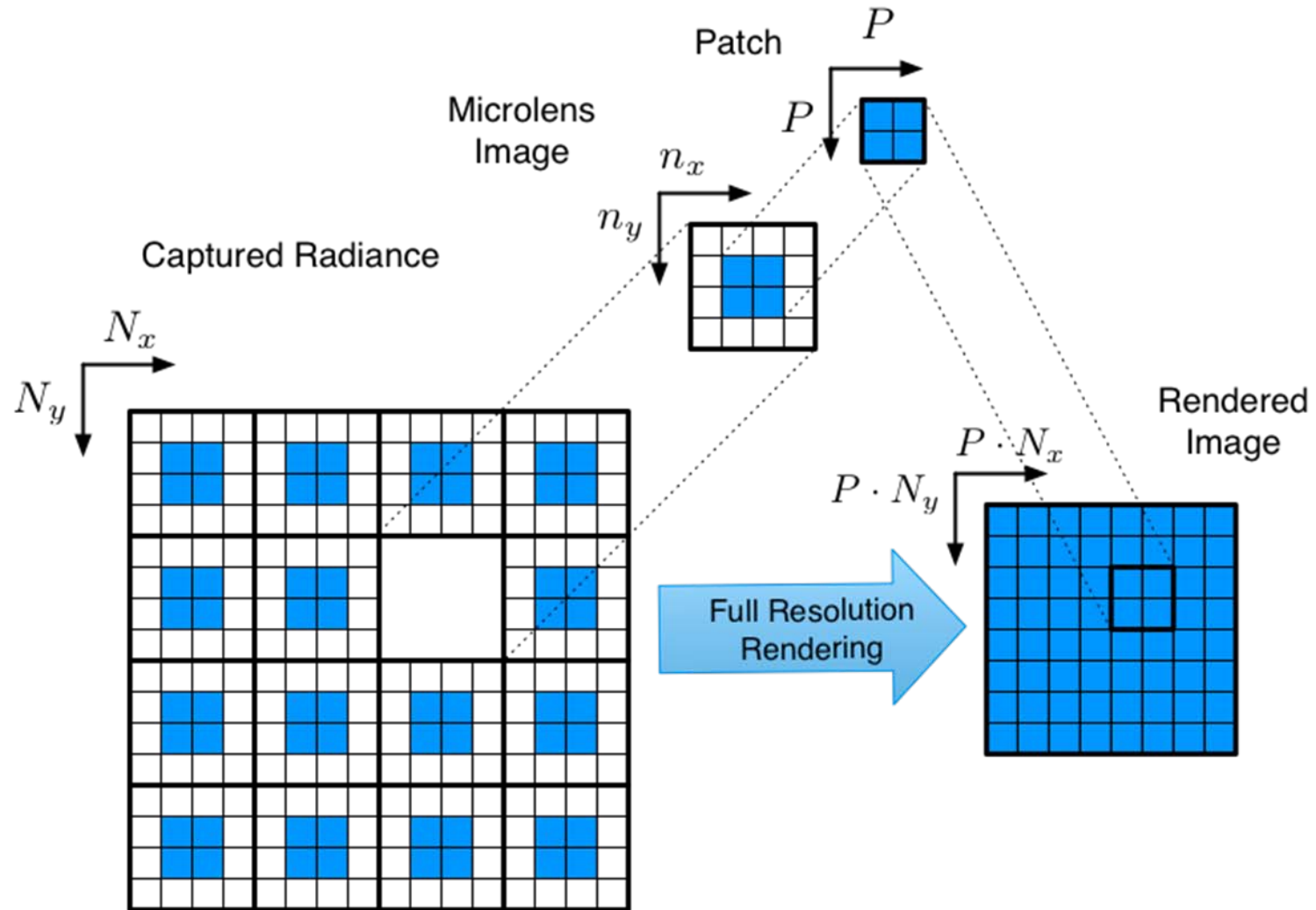
glActiveTexture(GL_TEXTURE0)
lfTexture = glGenTextures(1)
glBindTexture(GL_TEXTURE_RECTANGLE_ARB, lfTexture)
glTexImage2D(GL_TEXTURE_RECTANGLE_ARB, 0, 3,
             image.size[0], image.size[1], 0,
             GL_RGBA, GL_UNSIGNED_BYTE, str_image)
```



GLSL Implementation of Rendering



GLSL Implementation of Rendering





GLSL Implementation of Rendering

```
{
    vec2 p = floor(gl_TexCoord[0].st / mu);
    vec2 qp = (gl_TexCoord[0].st / blockSize - p) * M
               + 0.5*(mu - M);
    vec2 fx = p * mu + qp;

    gl_FragColor = texture2DRect(flat , fx);
}

{
    vec2 p = floor(gl_TexCoord[0].st / mu);
    vec2 qp = (gl_TexCoord[0].st / blockSize - p) * M
               + 0.5*(mu - M);
    vec2 fx = p * mu + qp;

    gl_FragColor = texture2DRect(flat , fx);
}
```

GLSL Implementation (Live Demo)

- ▶ Rendering
- ▶ Parallax
- ▶ Refocusing

Wrap Up

Questions answered
Issues explored

Questions?

References

Recommended Reading

- ▶ Reading material has been selected for its educational value
- ▶ Many important references have been dropped in the effort to make this reading list short



Reading Material

<http://www.cs.umd.edu/class/spring2005/cmsc828v/papers/p31-levoy.pdf>

<http://www.cs.utah.edu/classes/cs5610/handouts/lumigraph.pdf>

GERRARD A., BURCH J. M.: Introduction to matrix methods in optics.

GUILLEMIN V., STERNBERG S.: Symplectic techniques in physics.

<http://www.tgeorgiev.net/IntegralView.pdf>

<http://physics.nist.gov/Divisions/Div844/manual/studymanual.html>

http://en.wikipedia.org/wiki/Hamiltonian_mechanics

[http://en.wikipedia.org/wiki/Liouville's_theorem_\(Hamiltonian\)](http://en.wikipedia.org/wiki/Liouville's_theorem_(Hamiltonian))

<http://portal.acm.org/citation.cfm?id=344779.344932>

<http://portal.acm.org/citation.cfm?id=1073320>

IVES F., US Patent 725,567. (Year: 1903)

<http://www.merl.com/people/raskar/Mask/>

<http://www.tgeorgiev.net/Radiance/>

LIPPMANN, G., Epreuvesreversibles, J. Phys. 7, pp 821-825, (1908)



Reading Material

http://www.integralresource.org/Integral_History.pdf

http://web.mit.edu/persci/people/adelson/pub_pdfs/plenoptic.pdf

<http://graphics.stanford.edu/papers/lfcamera/>

<http://graphics.stanford.edu/papers/CameraArray/>

<http://www.tgeorgiev.net/Spatioangular.pdf>

And the web page of this course:

<http://www.tgeorgiev.net/Asia2009/>