

UNIVERSITY OF
CAMBRIDGE
COMPUTER LABORATORY



Advanced Graphics & Image Processing

Image-based rendering a.k.a. novel-view synthesis

Rafał Mantiuk

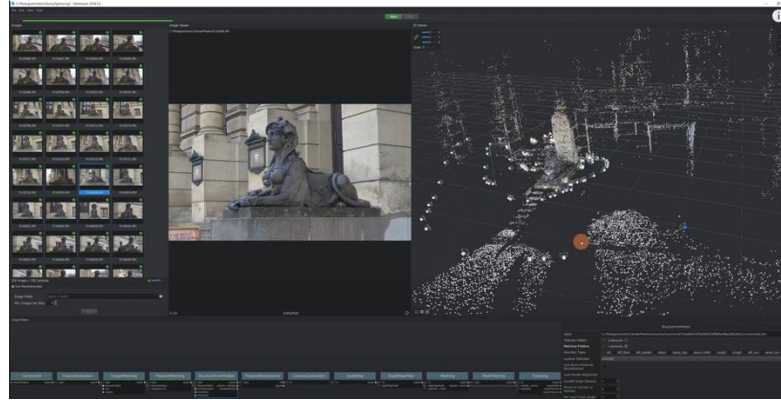
Computer Laboratory, University of Cambridge

What is image-based rendering (IBR)?

- ▶ IBR $\approx$ use images for 3D rendering



3D mesh + textures + shading



Photogrammetry

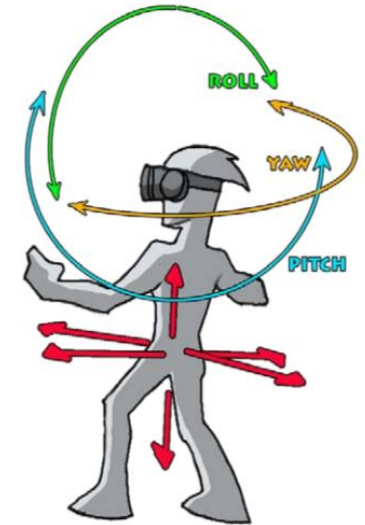


Neural Radiance Fields

- ▶ Our focus: methods that let us capture content with cameras

Motivation: why do we need image-based rendering?

- ▶ For inexpensive creation of high-quality 3D content
 - ▶ Minimize manual steps
 - ▶ Use cameras, which are good and abundant
- ▶ Why do we need 3D content?
 - ▶ AR/VR (+ novel display tech)
 - ▶ User-created content
 - ▶ 3D-printing
 - ▶ E-commerce



3D computer graphics

- ▶ We need:
 - ▶ Geometry + materials + textures
 - ▶ Lights
- ▶ Full control of illumination, realistic material appearance
- ▶ Graphics assets are expensive to create
- ▶ Rendering can be expensive
 - ▶ Shading tends to take most of the computation



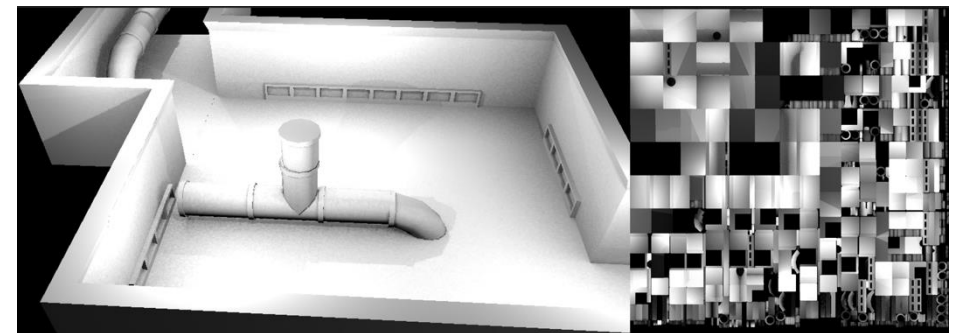
Cyberpunk 2077 (C) 2020 by CD Projekt RED

Baked / precomputed illumination

- ▶ We need:
 - ▶ Geometry + textures + (light maps)
- ▶ No need to scan and model materials
- ▶ Much faster rendering – simplified shading
- ▶ But we cannot change the illumination (relighting)



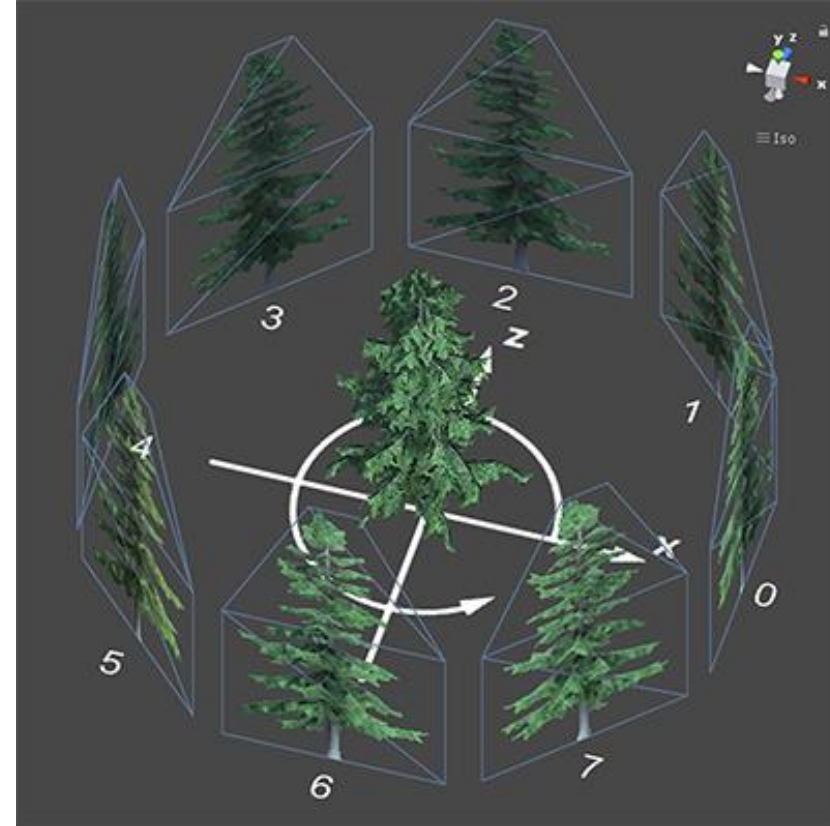
Google Earth



Precomputed light maps (from Wikipedia)

Billboards / Sprites

- ▶ We need:
 - ▶ Simplified geometry + textures (with alpha)
 - ▶ Lights
- ▶ Much faster to render than objects with 1000s of triangles
- ▶ Used for distant objects
 - ▶ or a small rendering budget
- ▶ Can be pre-computed from complex geometry



A tree rendered from a set of billboards

From:

<https://docs.unity3d.com/ScriptReference/BillboardAsset.html>

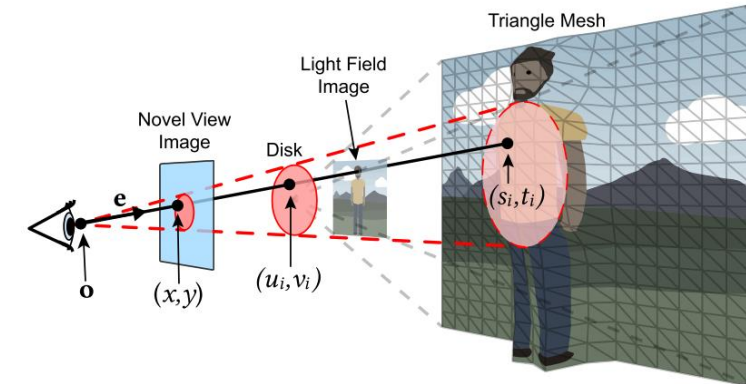
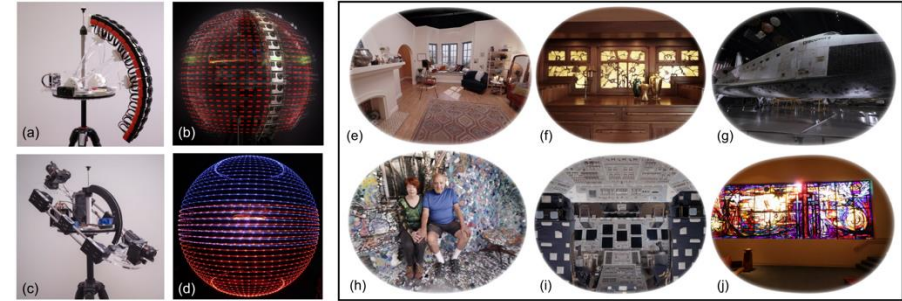
Light fields

- ▶ We need:
 - ▶ Images of the scene
 - ▶ Or a microlens image
- ▶ Does not need any geometry
 - ▶ But requires a large number of images for good quality
- ▶ Photographs are reprojected on a (focal) plane
- ▶ No relighting



Light fields + depth

- ▶ We need:
 - ▶ Depth map
 - ▶ Images of the object/scene
- ▶ We can use camera-captured images
- ▶ View-dependent shading
- ▶ Depth-map can be computed using multi-view stereo techniques
- ▶ No relighting

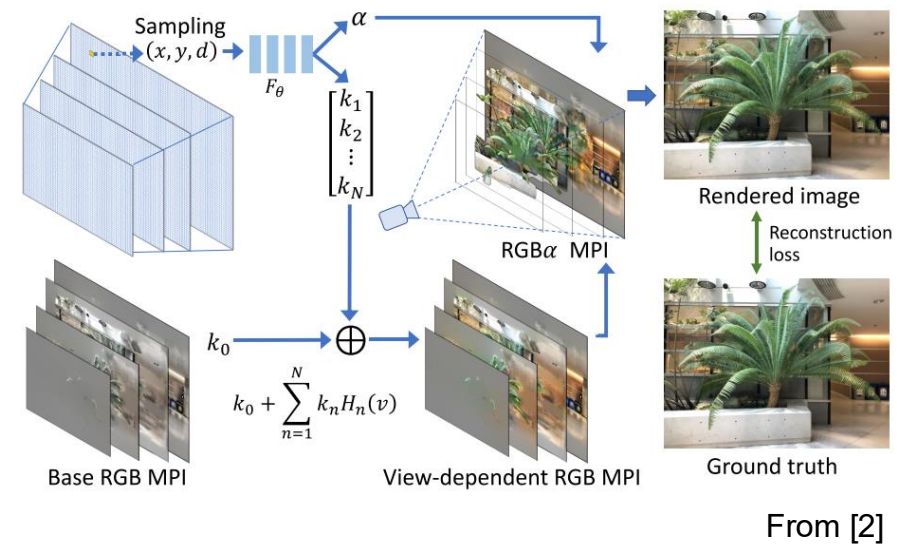
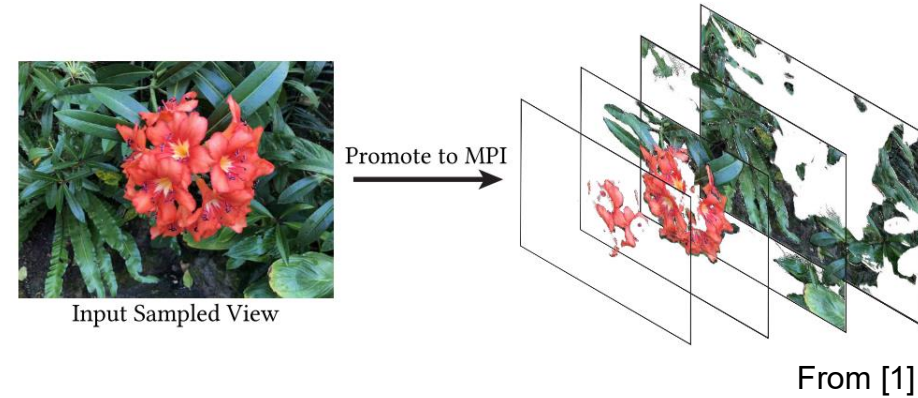


A depth map is approximated by a triangle mesh and rasterized. From: Overbeck et al. TOG 2018, <https://doi.org/10.1145/3272127.3275031>.

Demo: <https://augmentedperception.github.io/welcome-to-lightfields/>

Multi-plane images (MPI)

- ▶ We need:
 - ▶ Images of the scene + camera poses
- ▶ Each plane: RGB + alpha
 - ▶ Decomposition formulated as an optimization problem
 - ▶ Differential rendering
- ▶ Only front view



[1] Mildenhall, et al. "Local Light Field Fusion." *ACM Transactions on Graphics* 38, no. 4 (July 12, 2019): 1–14.

<https://doi.org/10.1145/3306346.3322980>

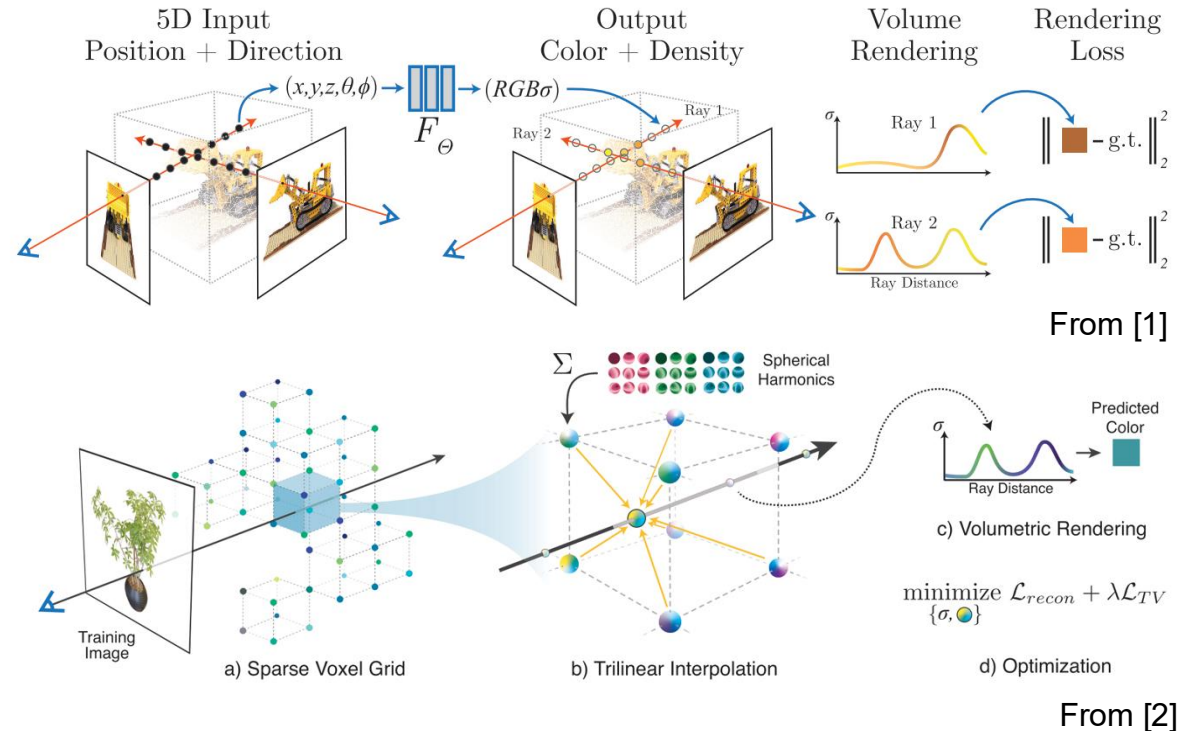
[2] Wizarwongsa et al. "NeX: Real-Time View Synthesis with Neural Basis Expansion." In *CVPR*, 8530–39. IEEE, 2021.

<https://doi.org/10.1109/CVPR46437.2021.00843>

<https://nex-mpi.github.io/>

Neural Radiance Fields (NeRF)

- ▶ We need
 - ▶ Images of the scene + camera poses
- ▶ Similar to MPI but stored in a volumetric data structure
 - ▶ Implicit: multi-layer perceptron
 - ▶ Explicit: Voxel grid
- ▶ Volumetric differential rendering



[1] Mildenhall, et al. "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis," 405–21, 2020. https://doi.org/10.1007/978-3-030-58452-8_24.

[2] Yu et al. "Plenoxels: Radiance Fields without Neural Networks." In *CVPR*, 5501–10, 2022. <http://arxiv.org/abs/2112.05131>.

UNIVERSITY OF
CAMBRIDGE
COMPUTER LABORATORY



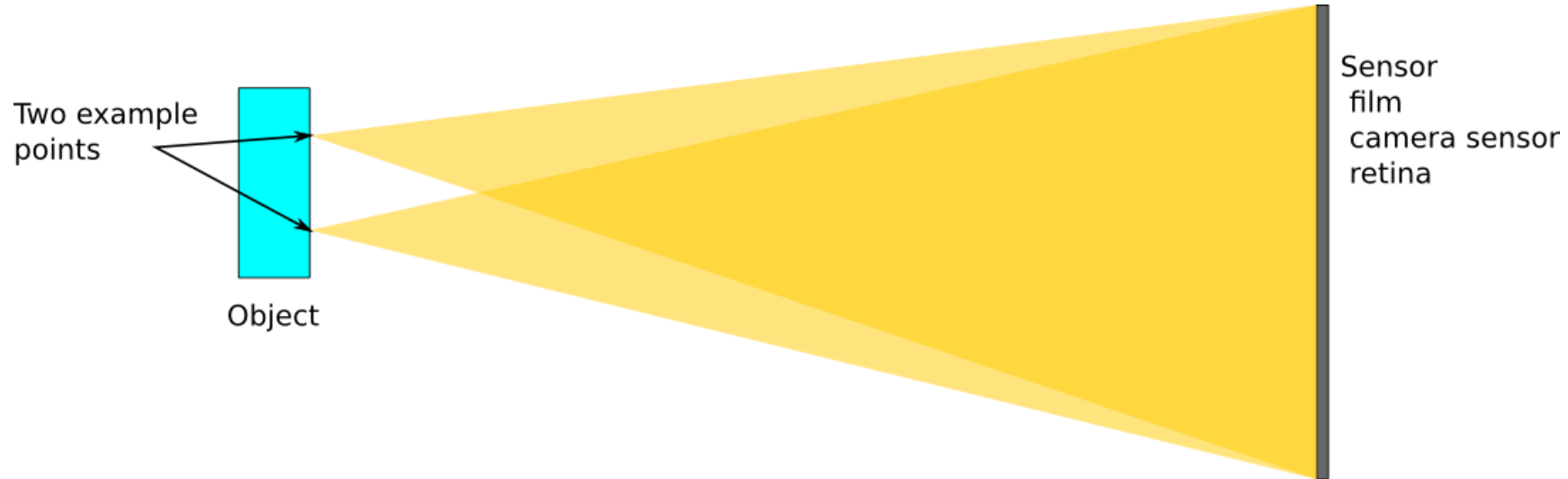
Finite aperture imaging

imaging and lens

Rafał Mantiuk

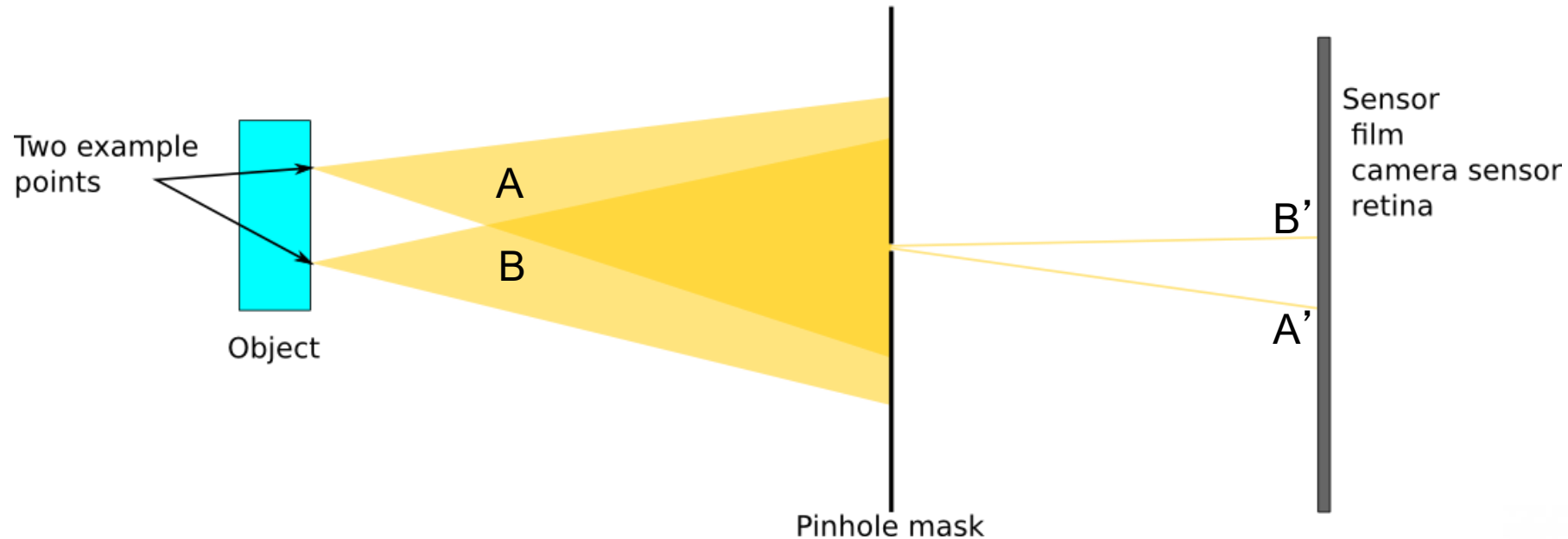
Computer Laboratory, University of Cambridge

Imaging – without lens



Every point in the scene illuminates every point (pixel) on a sensor. Everything overlaps - no useful image.

Imaging – pinhole camera

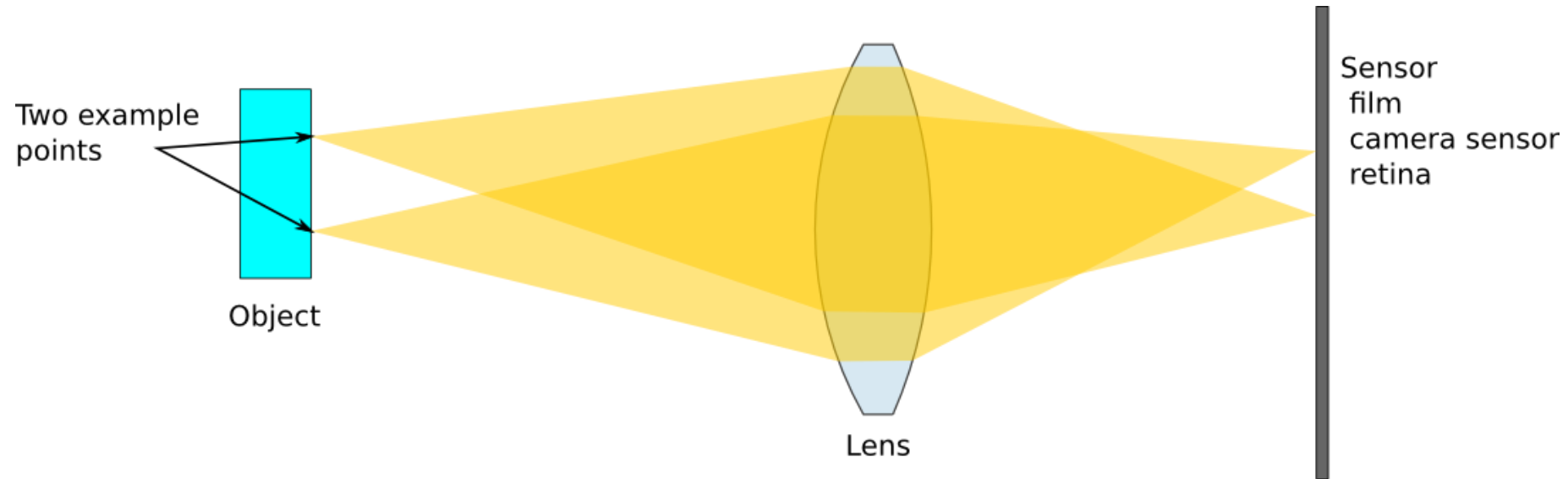


Pinhole masks all but only tiny beams of light. The light from different points is separated and the image is formed.

But very little light reaches the sensor.



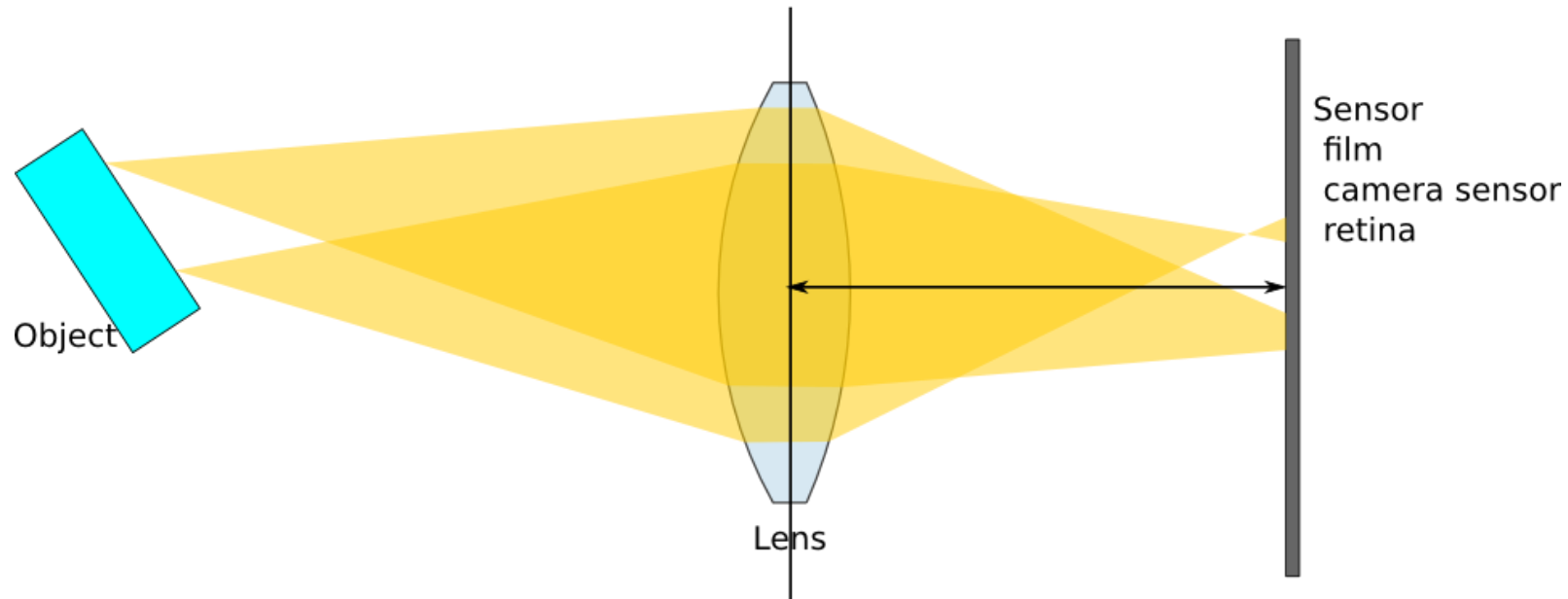
Imaging – lens



A lens can focus a beam of light on a sensor (focal plane).

Much more light-efficient than the pinhole.

Imaging – lens

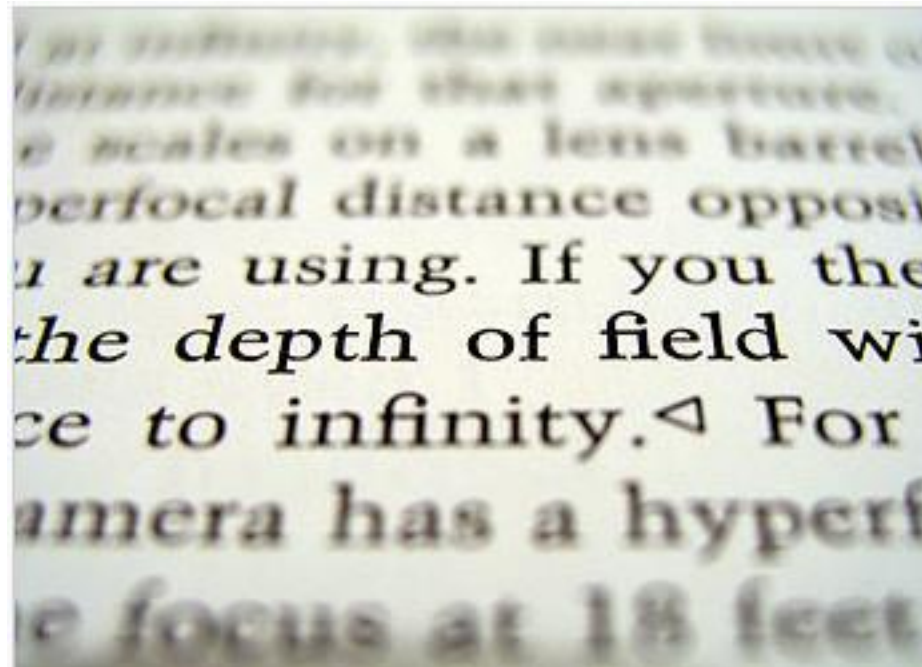


But if the light beams coming from different distances are not focused on the same plane. These points will appear blurry in the resulting image.

The camera needs to move the lens to focus an image on the sensor.

Depth of field

- ▶ Depth of field – range of depths that provides sufficient focus



Defocus blur is often desirable

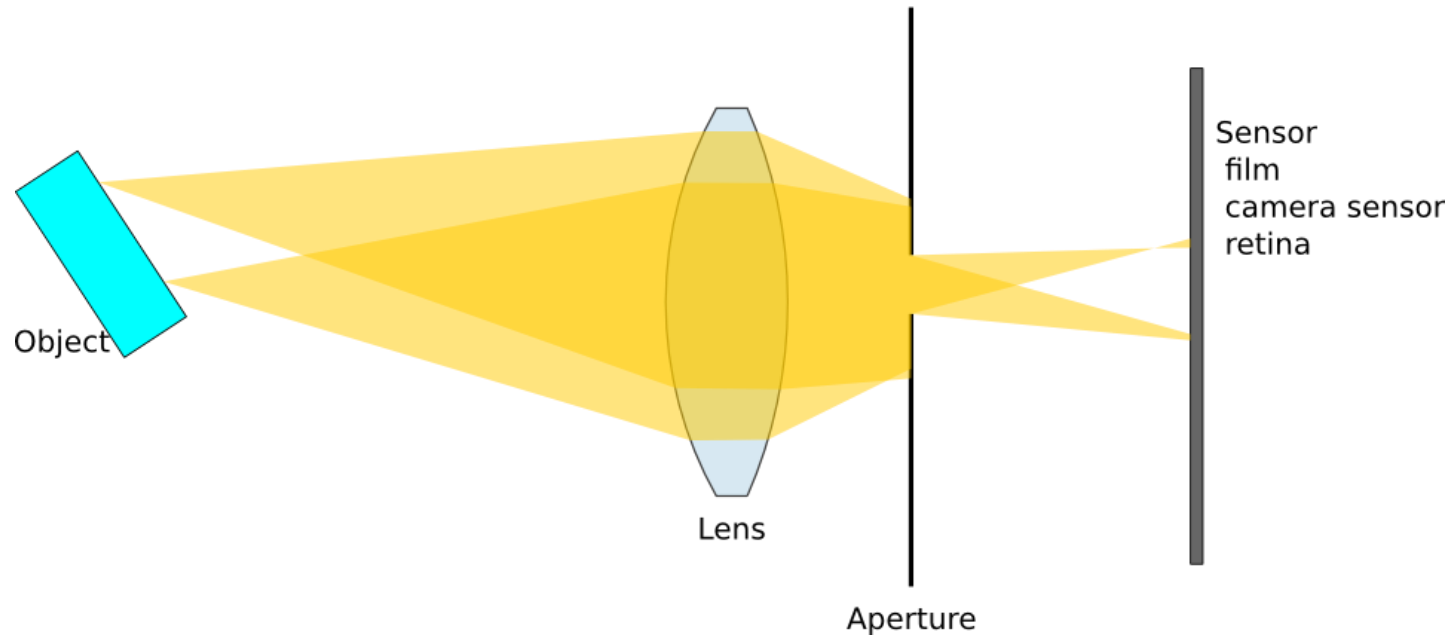


To separate the object of interest from background



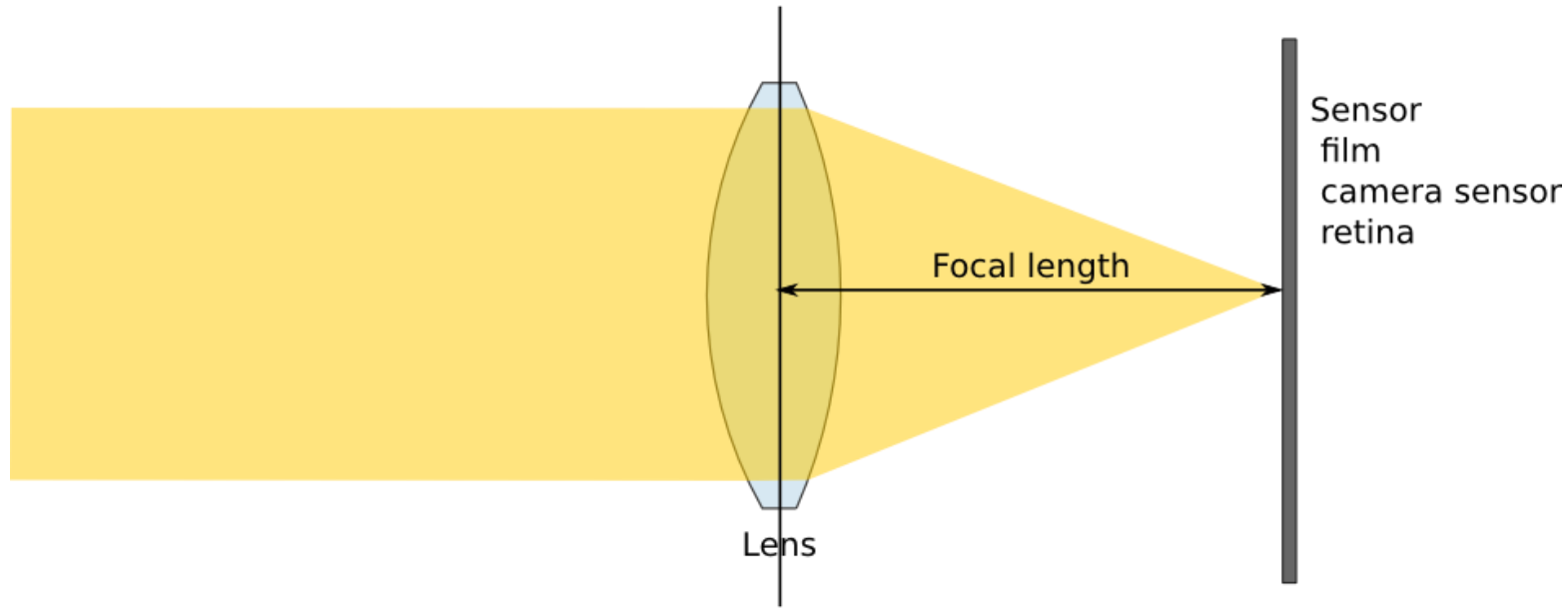
Defocus blur is a strong depth cue

Imaging – aperture



Aperture (introduced behind the lens) reduces the amount of light reaching the sensor, but it also reduces blurriness from defocus (increases depth-of-field).

Imaging – lens



Focal length – the distance between the sensor and the lens that is needed to focus light coming from an infinite distance.

Larger focal length of a lens – more or less magnification?

UNIVERSITY OF
CAMBRIDGE
COMPUTER LABORATORY



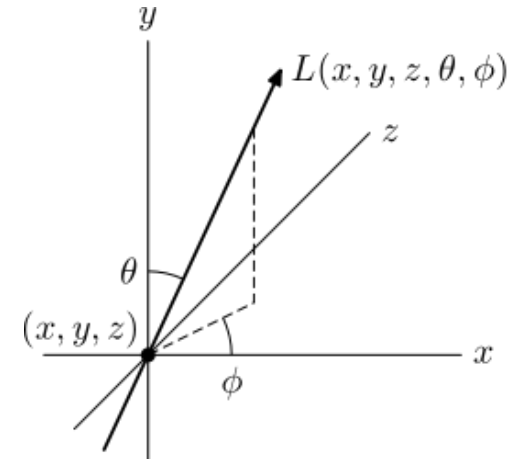
Light fields

Rafał Mantiuk

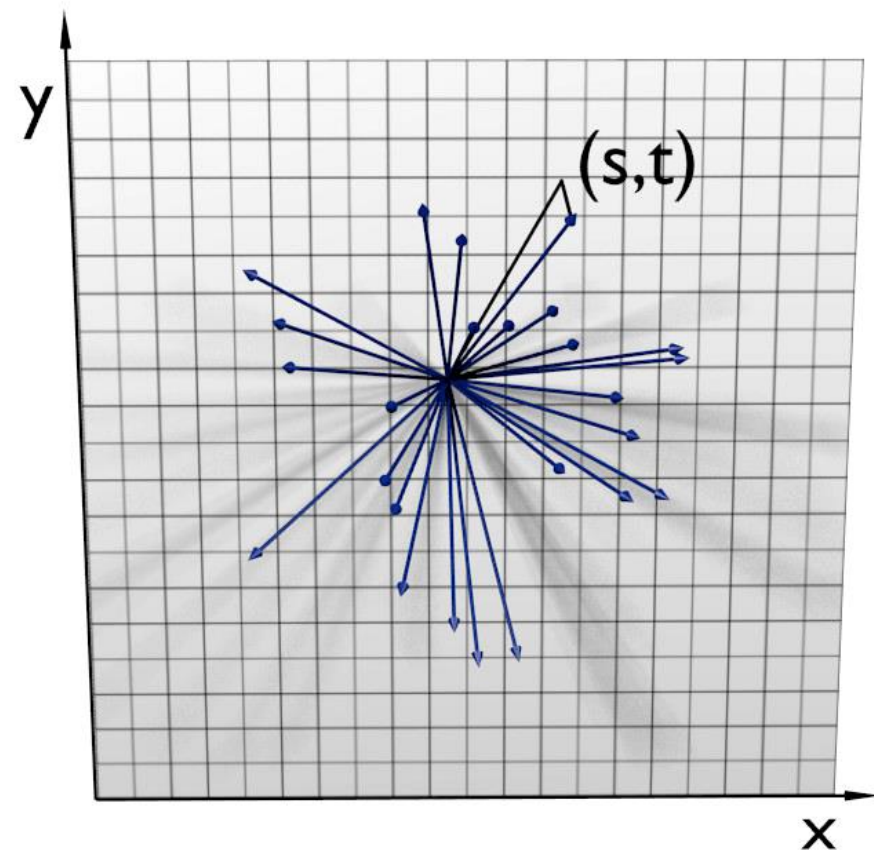
Computer Laboratory, University of Cambridge

From a plenoptic function to a light field

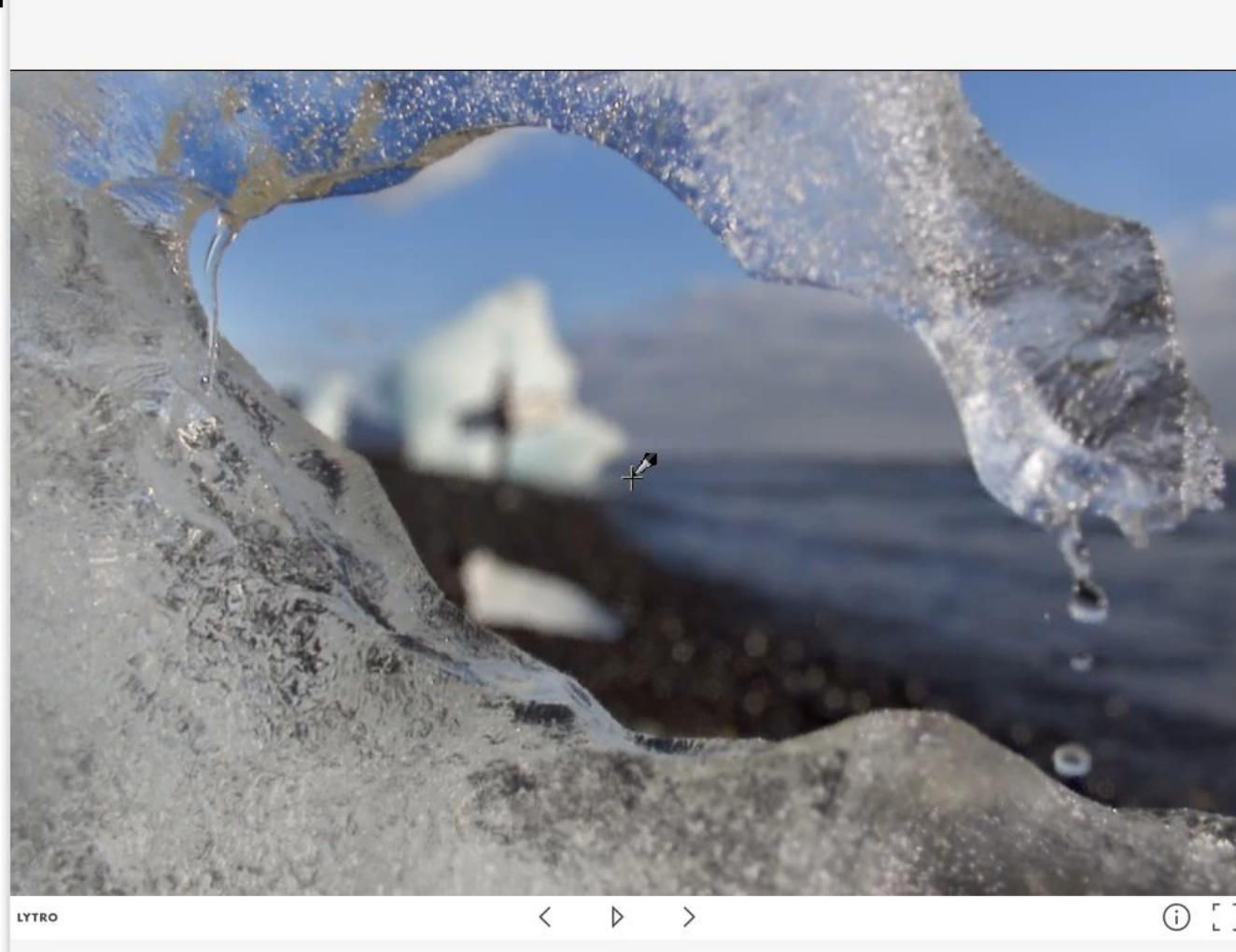
- ▶ Plenoptic function – describes all possible rays in a 3D space
 - ▶ Function of position (x, y, z) and ray direction (θ, ϕ)
 - ▶ But also wavelength λ and time t
 - ▶ Between 5 and 7 dimensions
- ▶ But the number of dimensions can be reduced if
 - ▶ The camera stays outside the convex hull of the object
 - ▶ The light travels in uniform medium
 - ▶ Then, radiance L remains the same along the ray (until the ray hits an object)
 - ▶ This way we obtain a **4D light field**



Planar 4D light field



Refocusing and view point adjustment

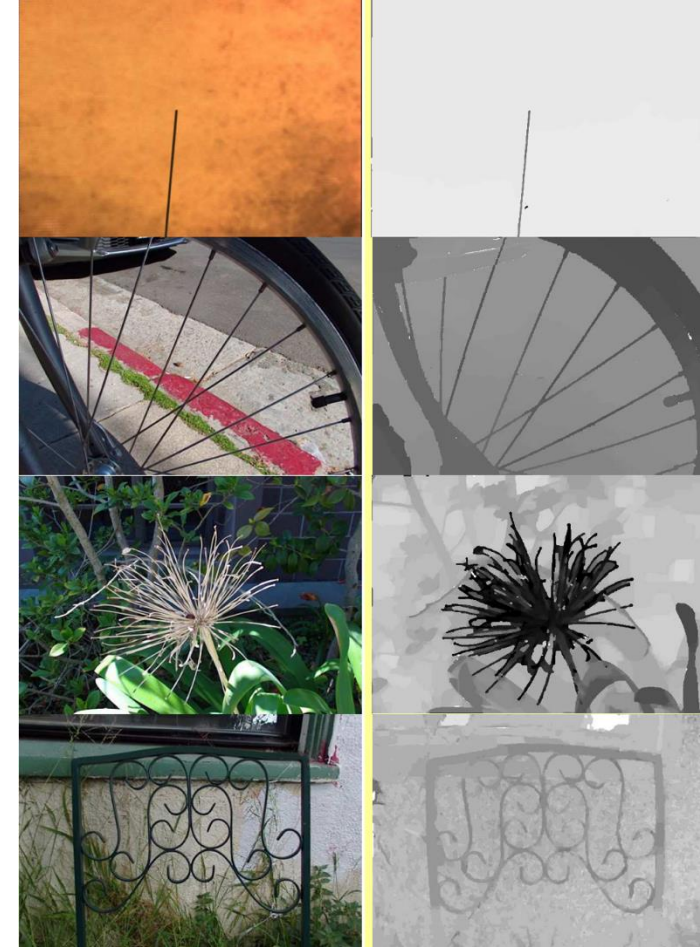


Depth estimation from light field

- ▶ Passive sensing of depth
- ▶ Light field captures multiple depth cues
 - ▶ Correspondance (disparity) between the views
 - ▶ Defocus
 - ▶ Occlusions

Central view

Reconstructed depth



From: *Ting-Chun Wang, Alexei A. Efros, Ravi Ramamoorthi*; The IEEE International Conference on Computer Vision (ICCV), 2015, pp. 3487-3495

Two methods to capture light fields

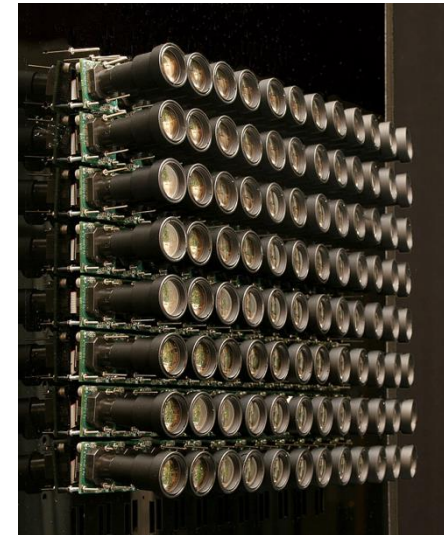
Micro-lens array

- ▶ Small baseline
- ▶ Good for digital refocusing
- ▶ Limited resolution

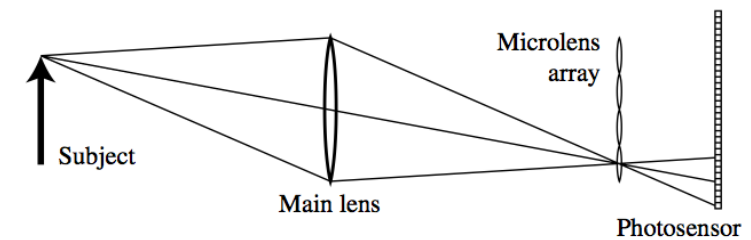
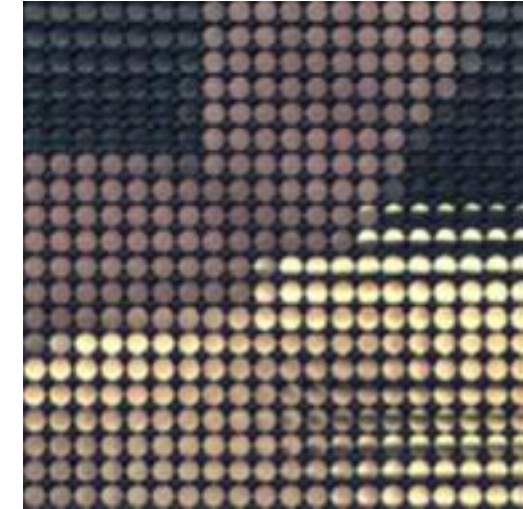


Camera array

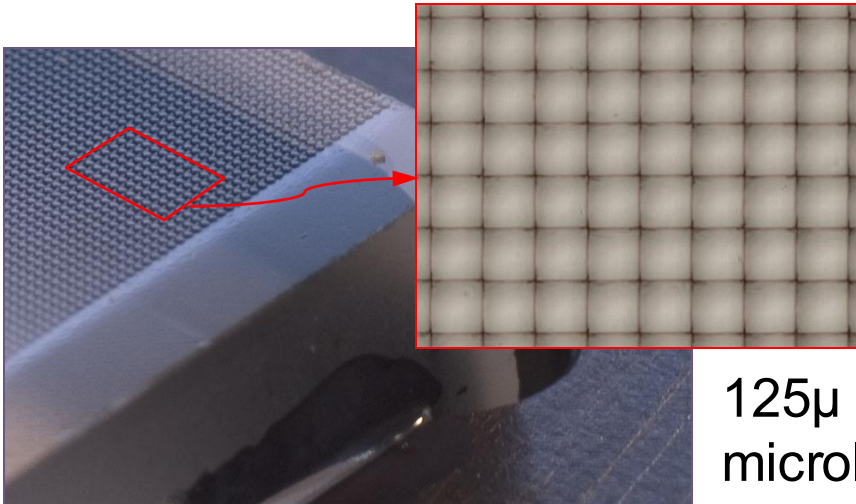
- ▶ Large baseline
- ▶ High resolution
- ▶ Rendering often requires approximate depth



Light field image – with microlens array

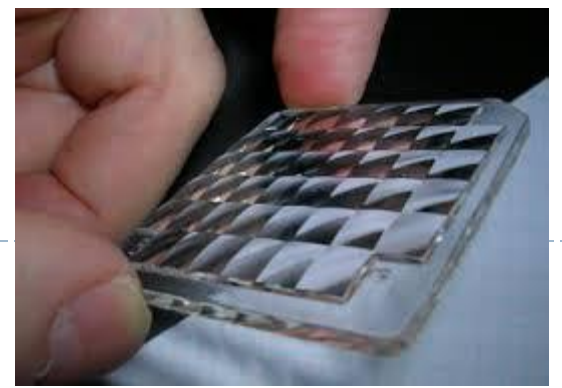


Digital Refocusing using Light Field Camera



125 μ square-sided
microlenses

Lenslet
array



[Ng et al 2005]

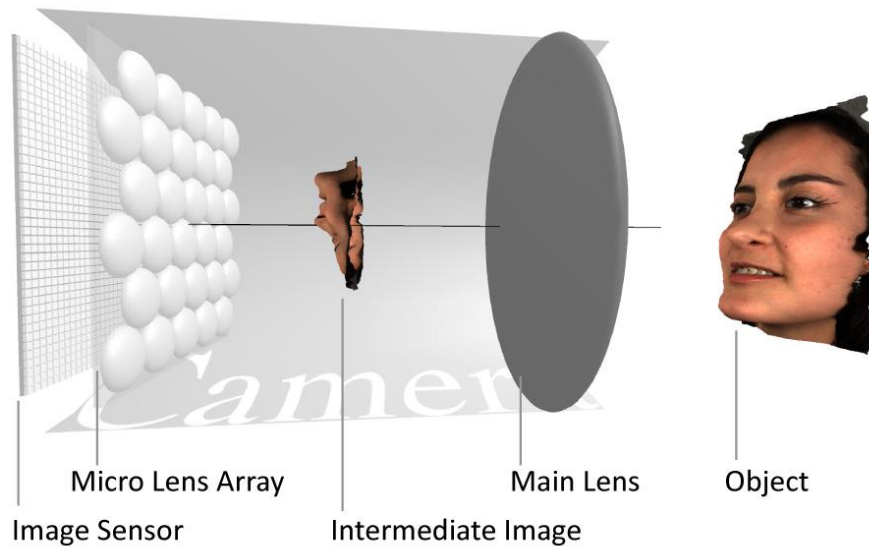
Lytro-cameras

- ▶ First commercial light-field cameras
- ▶ Lytro illum camera
 - ▶ 40 Mega-rays
 - ▶ 2D resolution: 2450 x 1634 (4 MPixels)

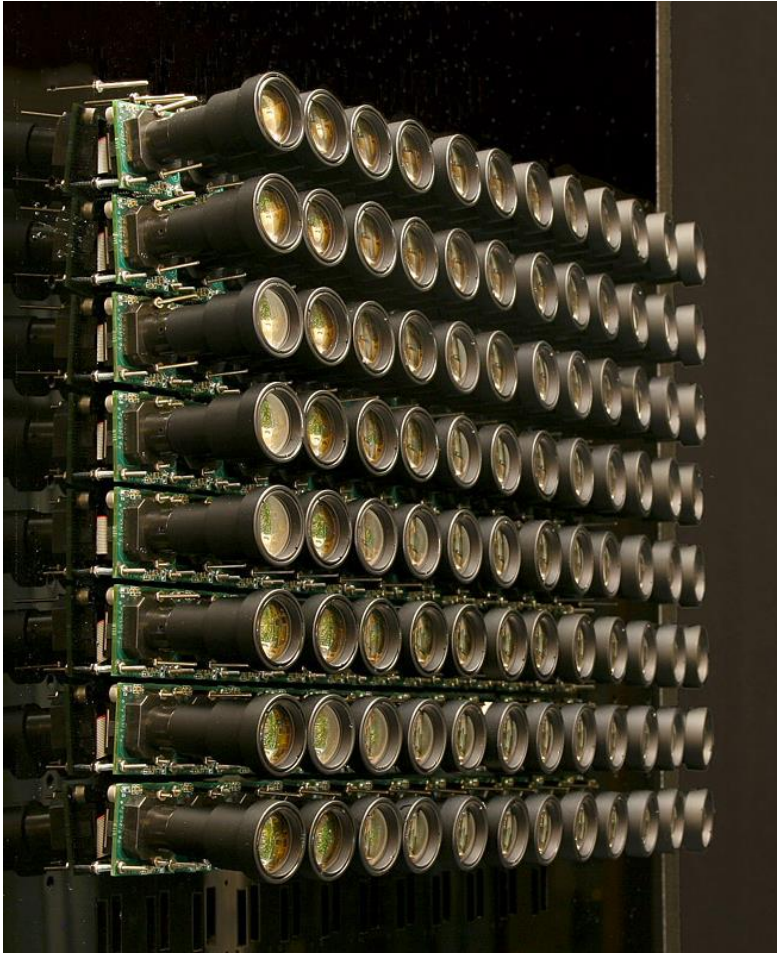


Raytrix camera

- ▶ Similar technology to Lytro
- ▶ But profiled for computer vision applications

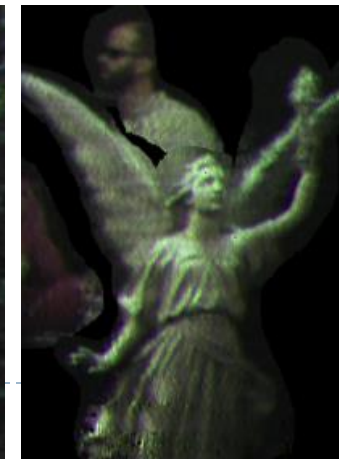
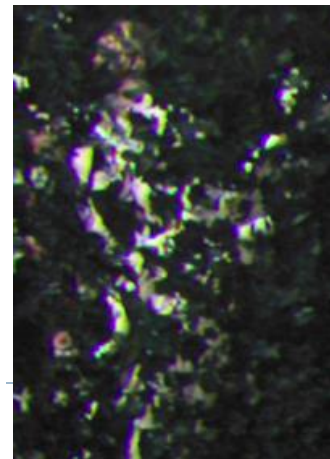


Stanford camera array



96 cameras

Application: Reconstruction of occluded surfaces



Seeing through forest

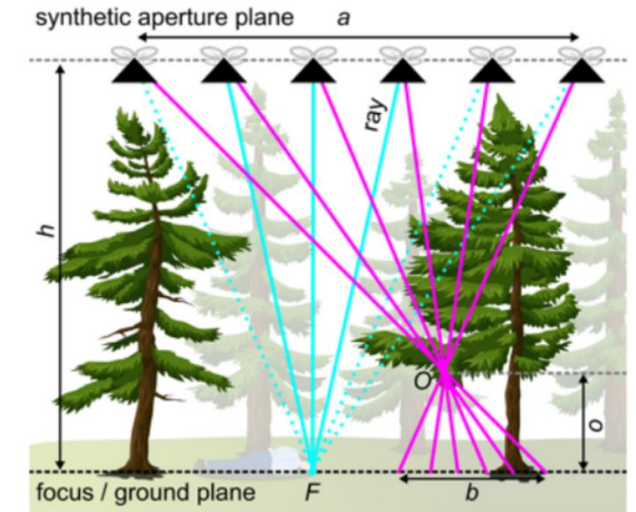
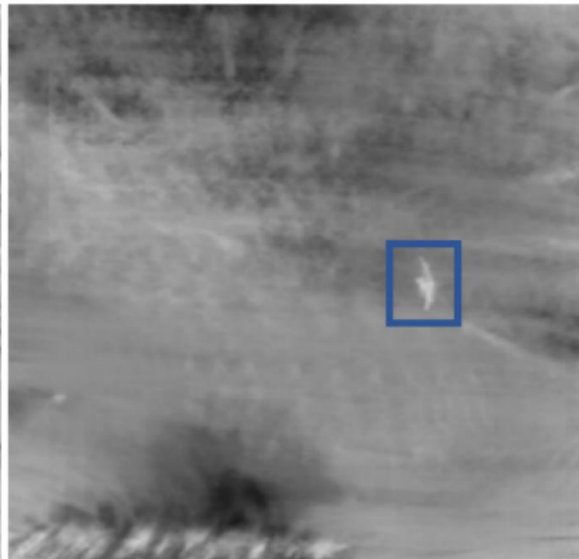
Optical synthetic aperture sensing

- ▶ A drone captures thermal images along the fly path
- ▶ A light field is reconstructed
- ▶ By refocusing on different depth, we can reduce occlusions

Thermal image, single view

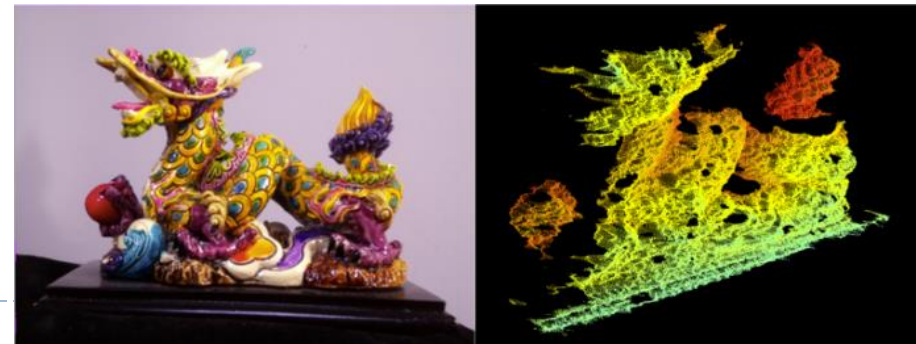
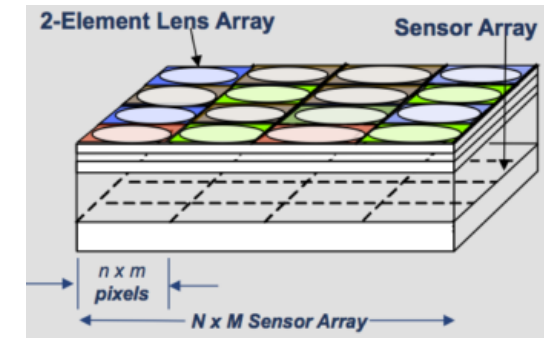
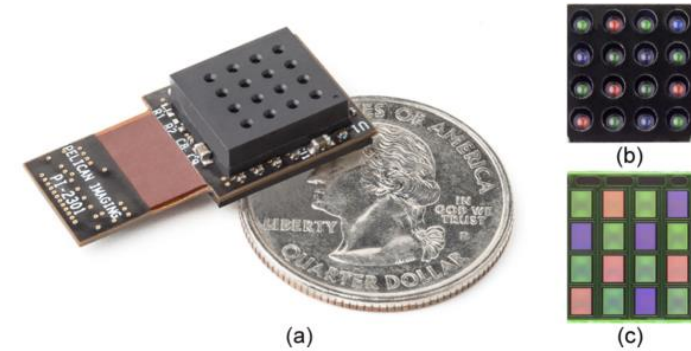


Synthetic large aperture image

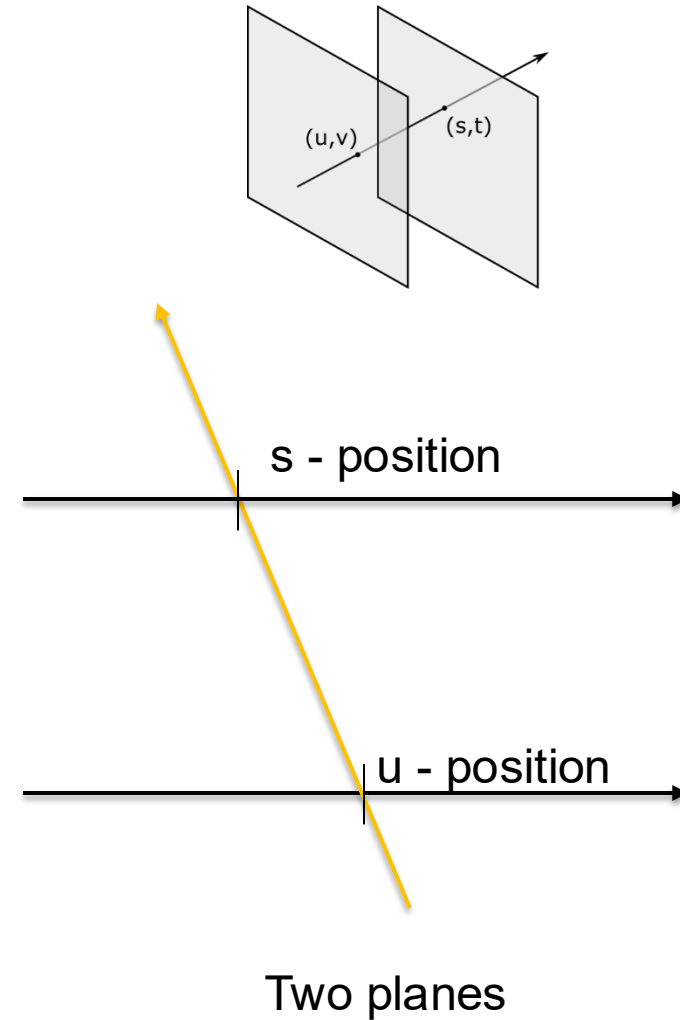
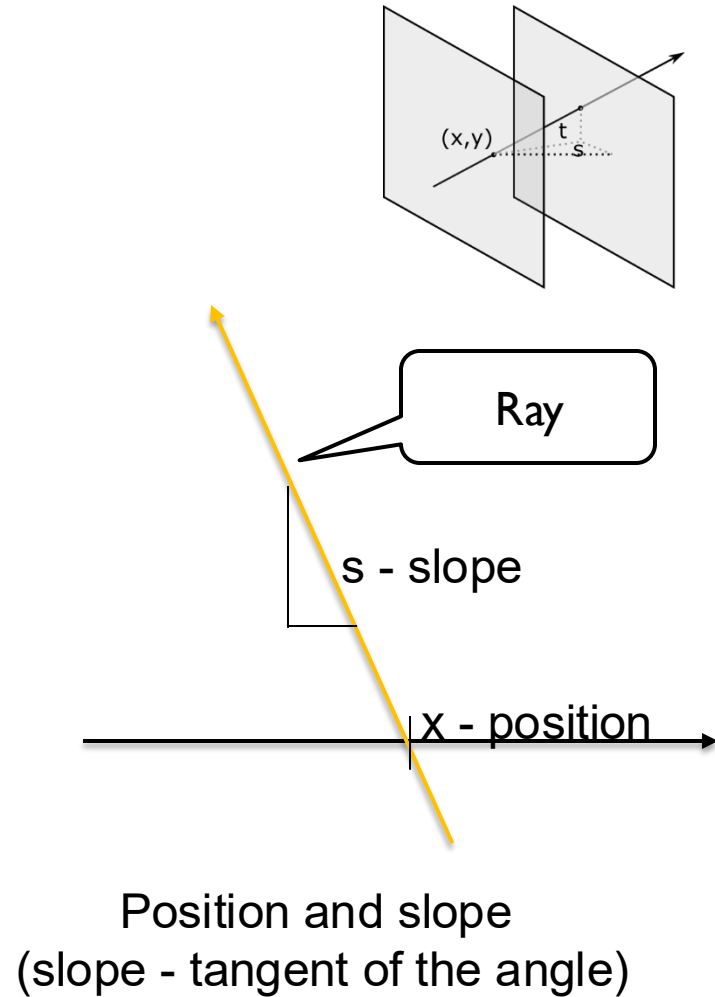


PiCam camera array module

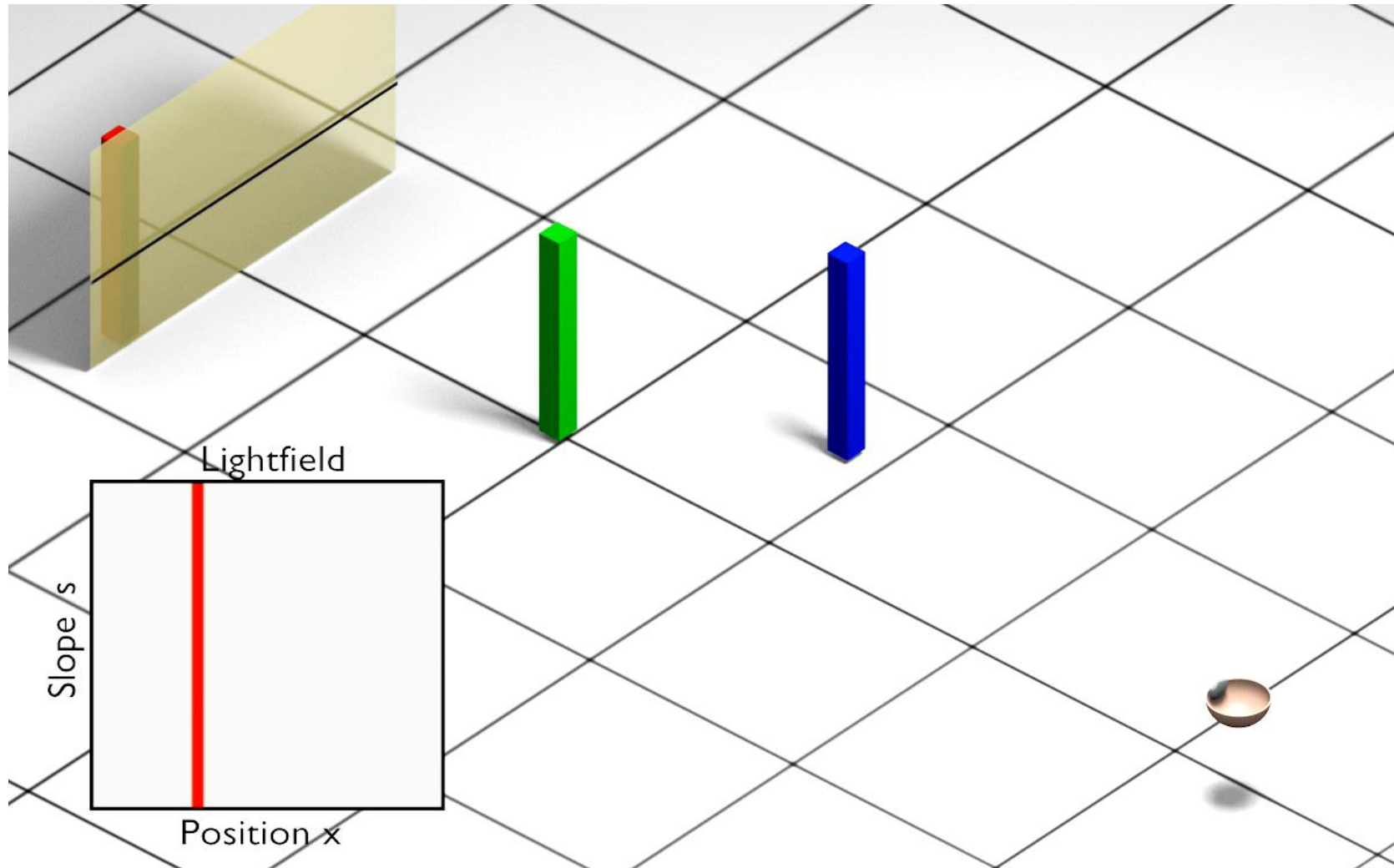
- ▶ Array of 4 x 4 cameras on a single chip
- ▶ Each camera has its own lens and senses only one spectral colour band
 - ▶ Optics can be optimized for that band
- ▶ The algorithm needs to reconstruct depth



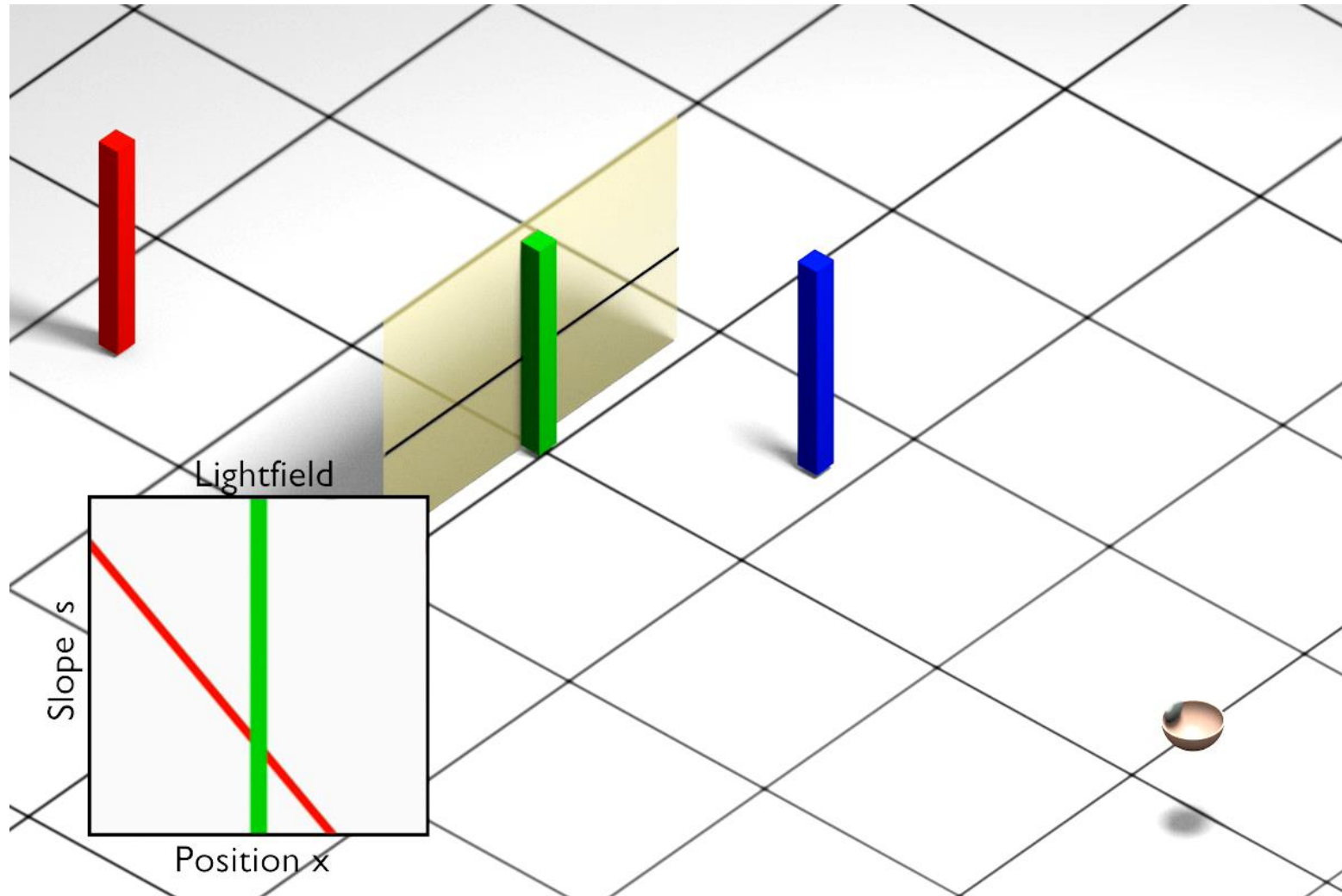
Light fields: two parametrisations (shown in 2D)



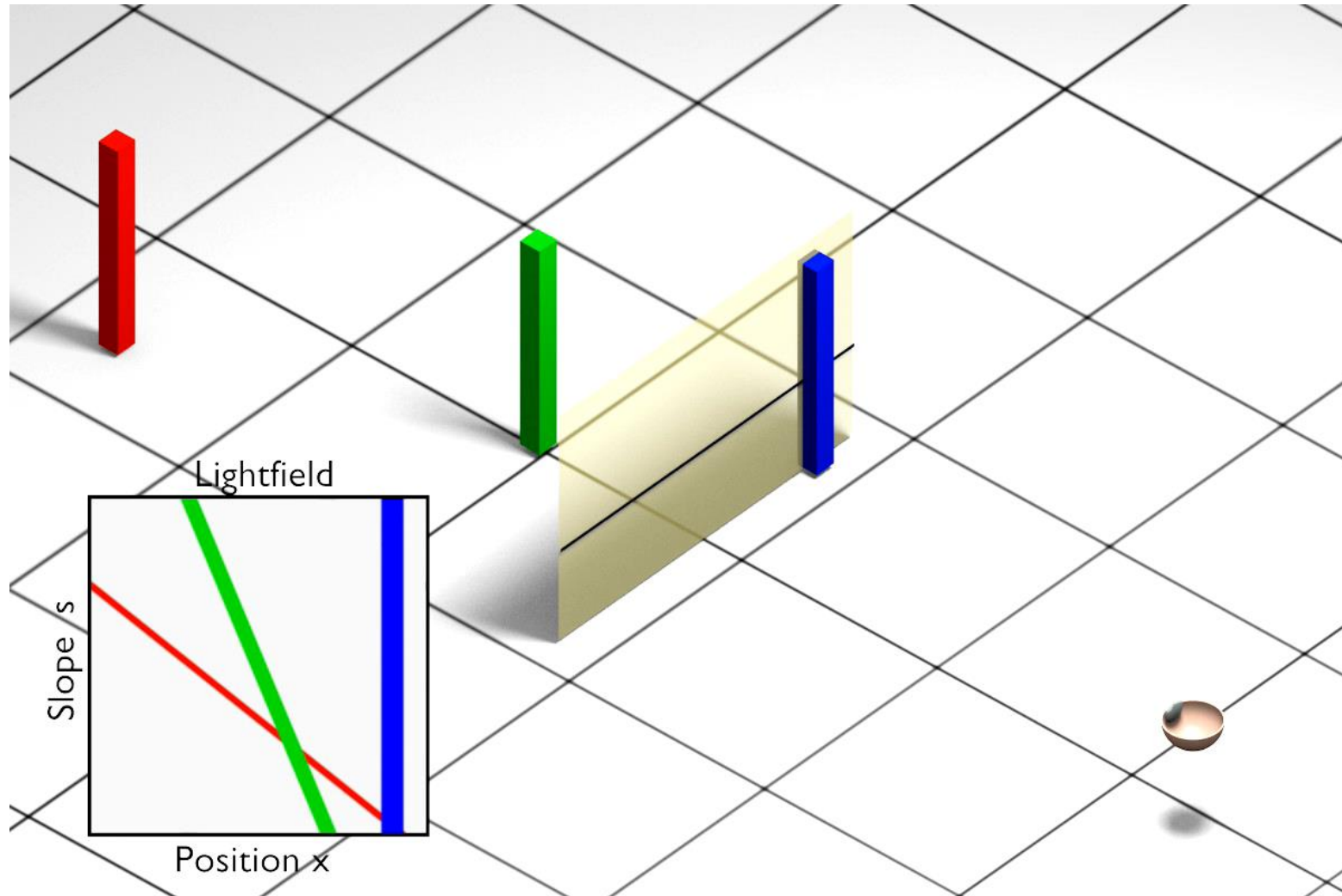
Lightfield - example



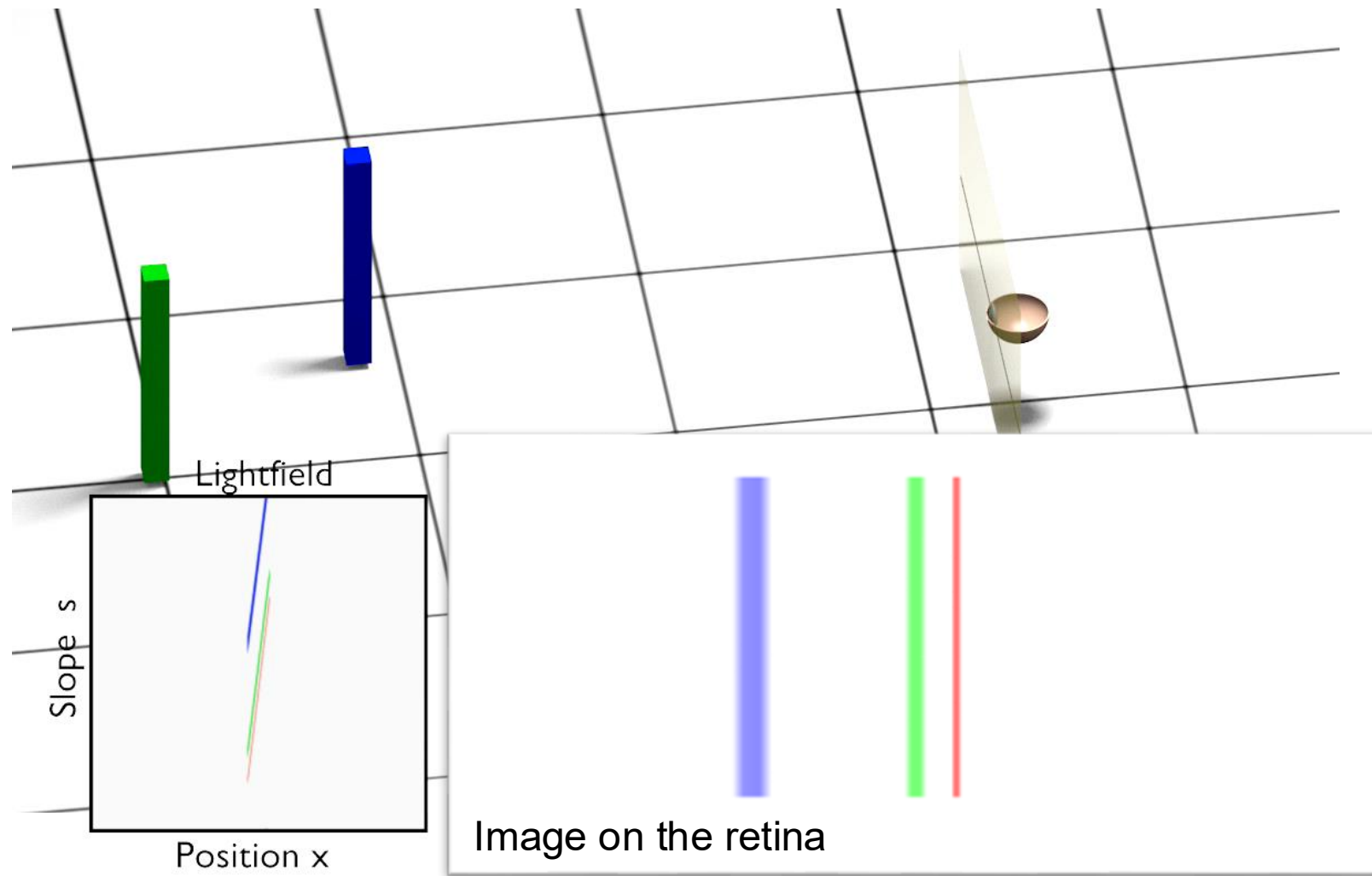
Lightfield - example



Lightfield - example



Lightfield - example



UNIVERSITY OF
CAMBRIDGE
COMPUTER LABORATORY

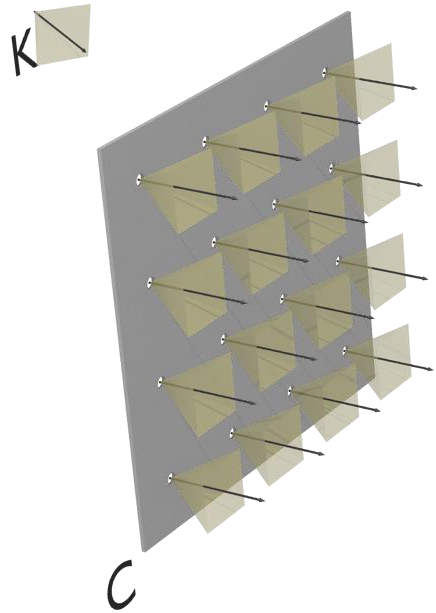


Light field rendering

Rafał Mantiuk

Computer Laboratory, University of Cambridge

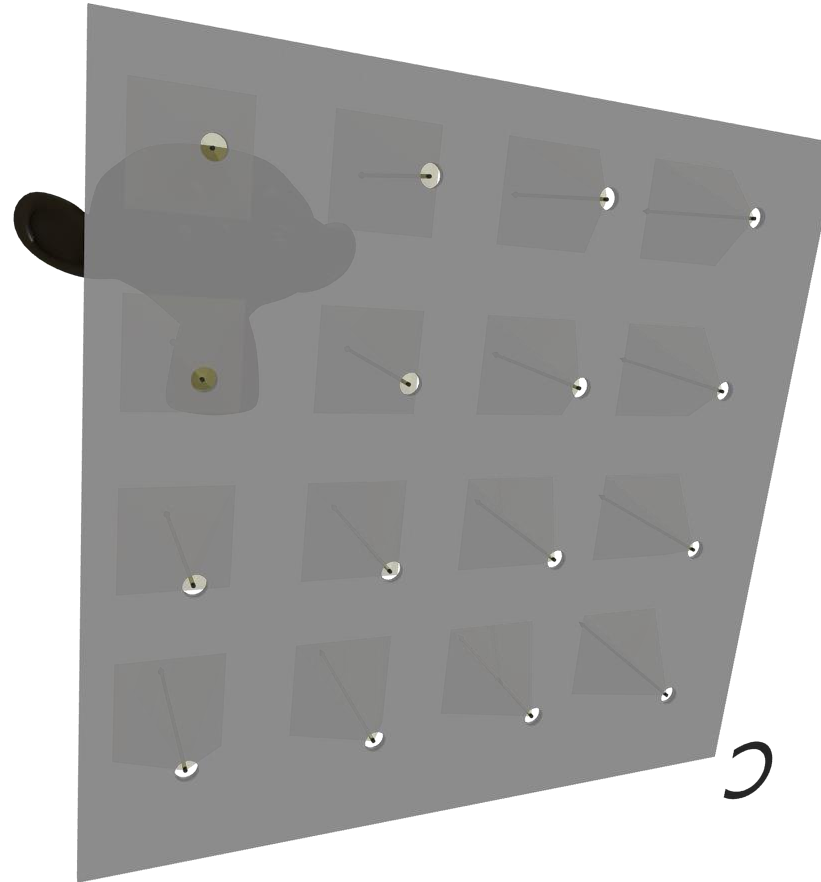
Light field rendering (1 / 3)



We want to render a scene (Blender monkey) as seen by camera K. We have a light field captured by a camera array. Each camera in the array has its aperture on plane C.

Light field rendering (2/3)

From the viewpoint of
camera K

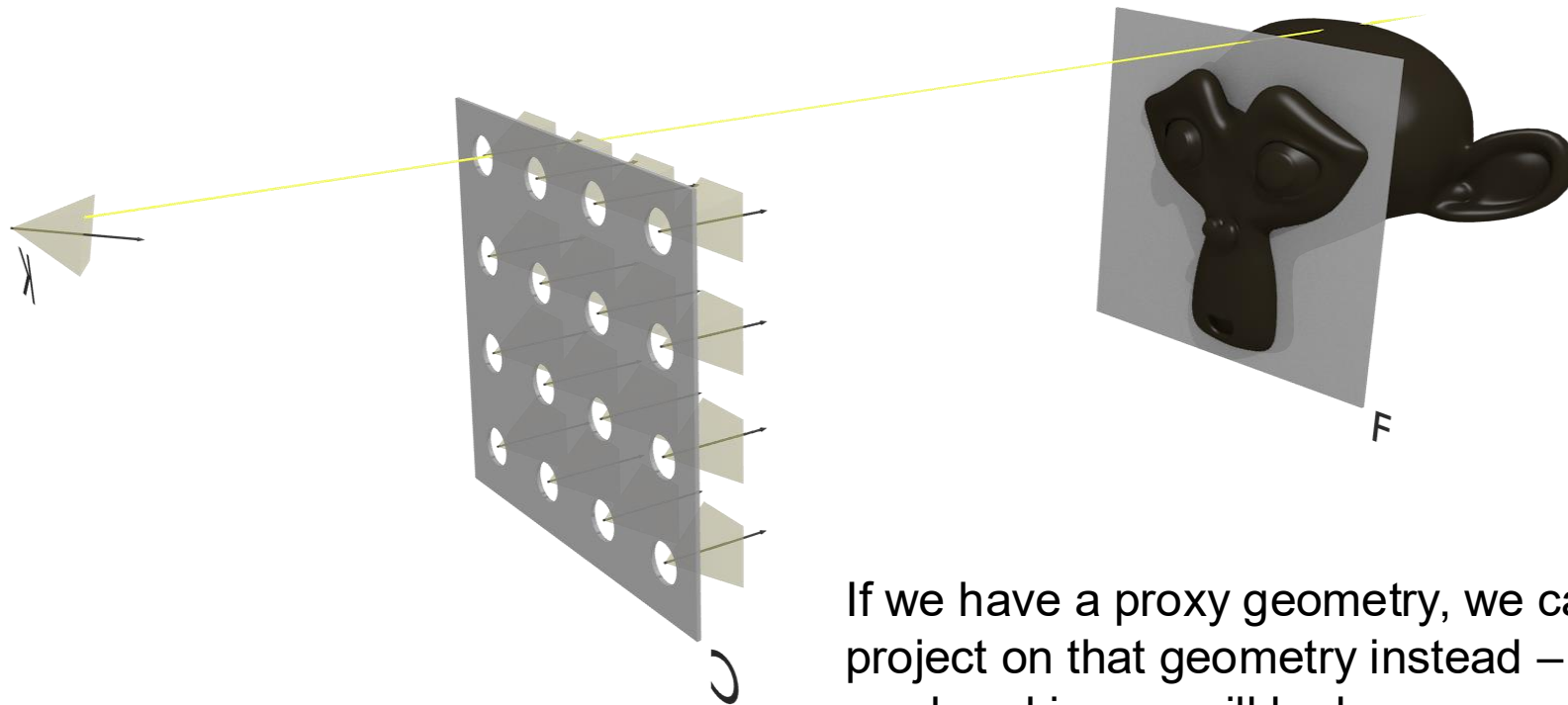


Each camera in the array provides accurate light measurements only for the rays originating from its pinhole aperture.

The missing rays can be either interpolated (reconstructed) or ignored.

Light field rendering (3/3)

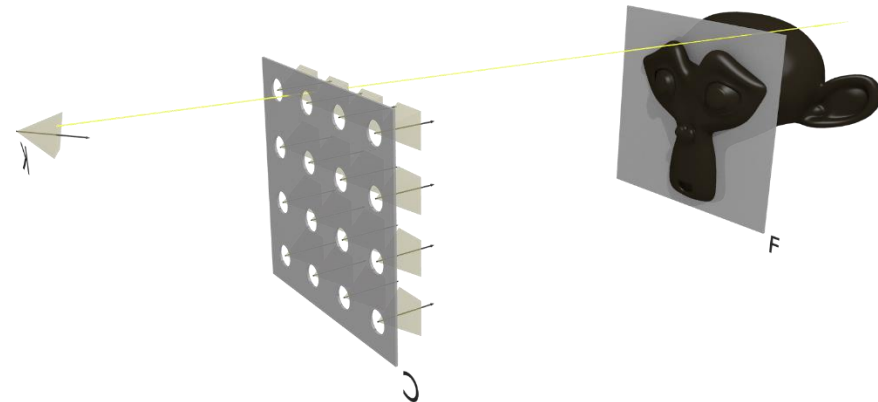
The rays from the camera need to be projected on the focal plane F . The objects on the focal plane will be sharp, and the objects in front or behind that plane will be blurry (ghosted), as in a traditional camera.



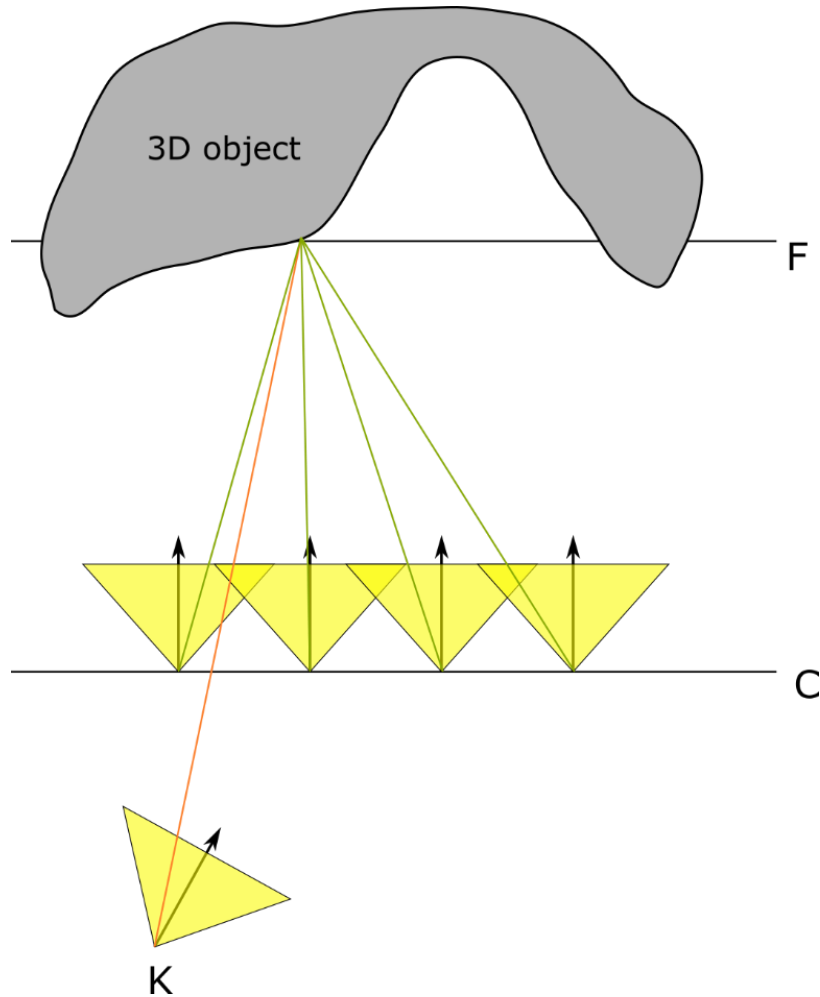
If we have a proxy geometry, we can project on that geometry instead – the rendered image will be less ghosted/blurry

Intuition behind light field rendering

- ▶ For large virtual aperture (use all cameras in the array)
 - ▶ Each camera in the array captures the scene
 - ▶ Then, each camera projects its image on the focal plane F
 - ▶ The virtual camera K captures the projection
- ▶ For small virtual aperture (pinhole)
 - ▶ For each ray from the virtual camera
 - ▶ interpolate rays from 4 nearest camera images
 - ▶ Or use the nearest-neighbour ray

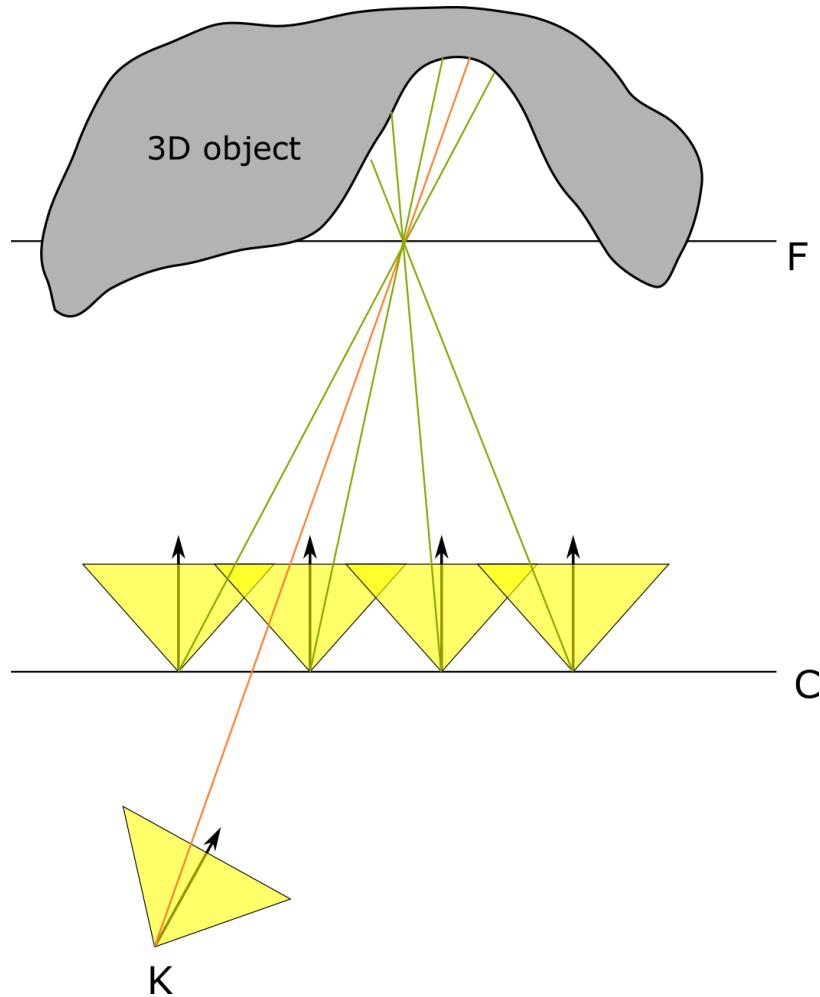


LF rendering – focal plane



- ▶ For a point on the focal plane, all cameras capture the same point on the 3D object
- ▶ They also capture approximately the same colour (for diffuse objects)
- ▶ Averaged colour will be the colour of the point on the surface

LF rendering – focal plane



- ▶ If the 3D object does not lie on the focal plane, all cameras capture different points on the object
- ▶ Averaging colour values will produce a „ghosted” image
- ▶ If we had unlimited number of cameras, this would produce a depth-of-field effect

Finding homographic transformation 1/3

- ▶ For the pixel coordinates p_k of the virtual camera K, we want to find the corresponding coordinates p_i in the camera array image
- ▶ Given the world 3D coordinates of a point w :

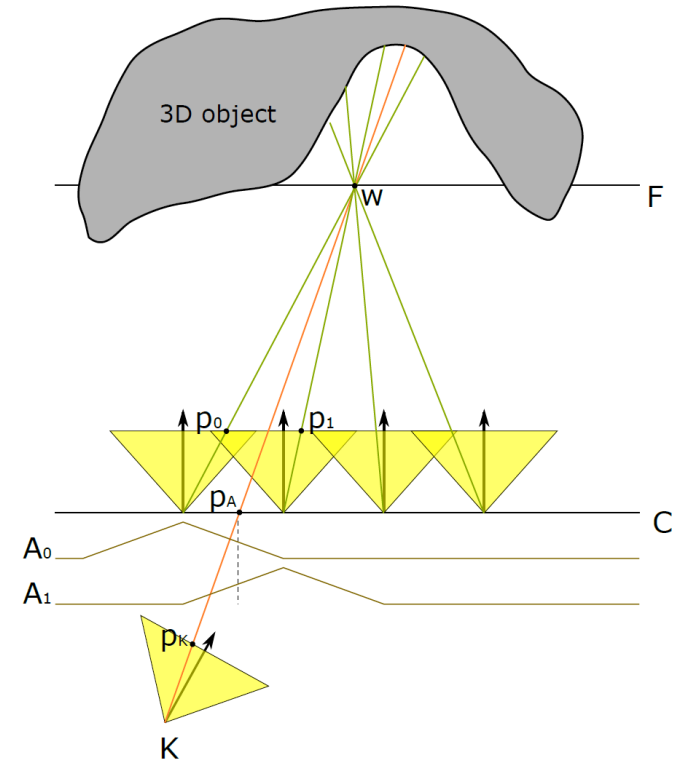
$$p_i = K P V_i w$$

Intrinsic
camera matrix

Projection
matrix

View
matrix

$$\begin{bmatrix} x_i \\ y_i \\ w_i \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} v_{11} & v_{12} & v_{13} & v_{14} \\ v_{21} & v_{22} & v_{23} & v_{24} \\ v_{31} & v_{32} & v_{33} & v_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$



Finding homographic transformation 2/3

- ▶ A homography between two views is usually found as:

$$\begin{aligned} \mathbf{p}_K &= \mathbf{K}_K \mathbf{P} \mathbf{V}_K \mathbf{w} \\ \mathbf{p}_i &= \mathbf{K}_i \mathbf{P} \mathbf{V}_i \mathbf{w} \end{aligned}$$

hence

$$\mathbf{p}_i = \mathbf{K}_i \mathbf{P} \mathbf{V}_i \mathbf{V}_K^{-1} \mathbf{P}^{-1} \mathbf{K}_K^{-1} \mathbf{p}_K$$

- ▶ But, $\mathbf{K}_K \mathbf{P} \mathbf{V}_K$ is not a square matrix and cannot be inverted
 - ▶ To find the correspondence, we need to constrain 3D coordinates $\mathbf{w}$ to lie on the plane:

$$\mathbf{N} \cdot (\mathbf{w} - \mathbf{w}_F) = 0 \quad \text{or} \quad d = \begin{bmatrix} n_x & n_y & n_z & -\mathbf{N} \cdot \mathbf{w}_F \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Finding homographic transformation 3/3

The plane in
the camera coordinates
(not the world coordinates)

- ▶ Then, we add the plane equation to the projection matrix

$$\begin{bmatrix} x_i \\ y_i \\ d_i \\ w_i \end{bmatrix} = \begin{bmatrix} f_x & 0 & 0 & c_x \\ 0 & f_y & 0 & c_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ n_x^{(c)} & n_y^{(c)} & n_z^{(c)} & -N^{(c)} \cdot w_F^{(c)} \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} v_{11} & v_{12} & v_{13} & v_{14} \\ v_{21} & v_{22} & v_{23} & v_{24} \\ v_{31} & v_{32} & v_{33} & v_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$\hat{p}_i \qquad \hat{K}_i \qquad \hat{P} \qquad V_i \qquad w$

- ▶ Where d_i is the distance to the plane
- ▶ Hence

$$\hat{p}_i = \hat{K}_i \hat{P} V_i V_K^{-1} \hat{P}^{-1} \hat{K}_K^{-1} p_K$$

UNIVERSITY OF
CAMBRIDGE
COMPUTER LABORATORY



Neural radiance fields

Differentiable volumetric rendering

Rafał Mantiuk

Computer Laboratory, University of Cambridge

Stereo magnification: learning view synthesis using multiplane images

- Synthesize motion parallax from two (stereo) views

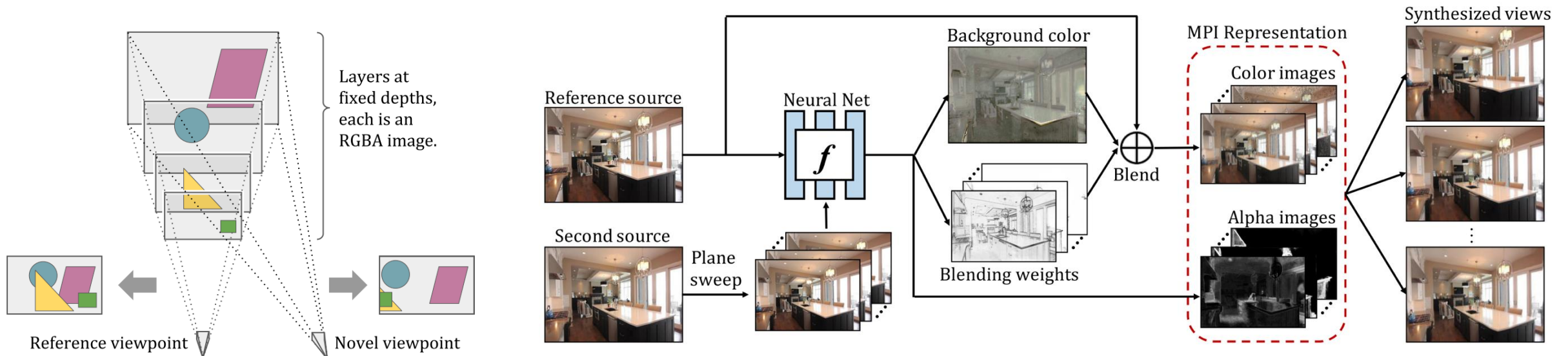
Left input image



Zhou, Tinghui, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. "Stereo Magnification: Learning View Synthesis Using Multiplane Images." *ACM Transactions on Graphics* 37, no. 4 (August 31, 2018): 1–12. <https://doi.org/10.1145/3197517.3201323>.

Stereo magnification: learning view synthesis using multiplane images

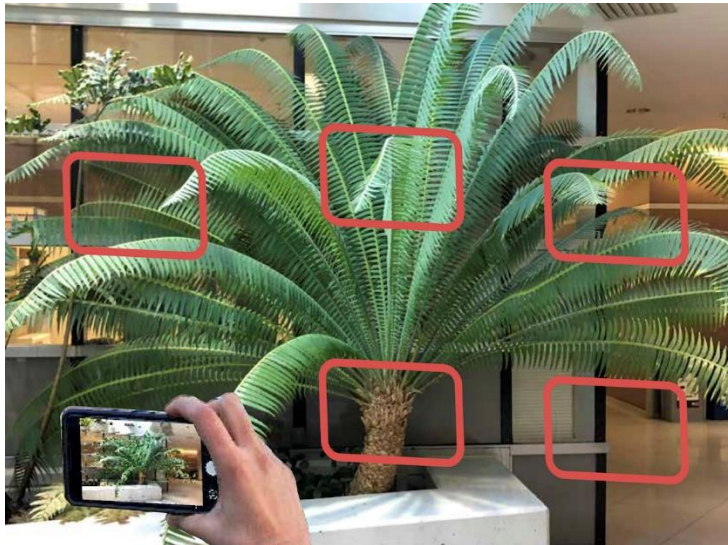
- ▶ Goal: decompose images into multiple planes with an alpha channel (MPI)
- ▶ Intermediate representation: background and foreground images
 - ▶ To better handle occlusions
 - ▶ The network is overfitted to each scene



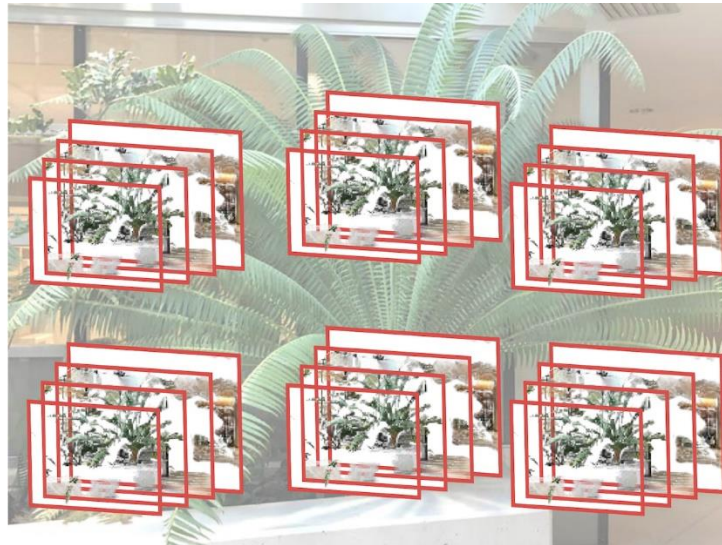
Local Light Field Fusion: Practical View Synthesis with Prescriptive Sampling Guidelines

- ▶ Reconstruct multiple MPs, then blend them
- ▶ This is to better capture view-dependent effects
 - ▶ E.g. specular reflections

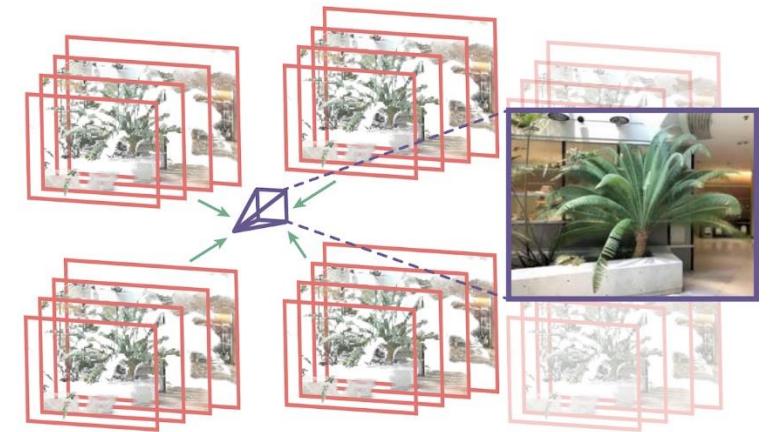
Mildenhall, Ben et al. “Local Light Field Fusion: Practical View Synthesis with Prescriptive Sampling Guidelines.” *ACM Trans. on Graphics* 38, no. 4 (July 12, 2019): 1–14.
<https://doi.org/10.1145/3306346.3322980>.



Fast and easy handheld capture with guideline: closest object moves at most D pixels between views



Promote sampled views to local light field via layered scene representation



Blend neighboring local light fields to render novel views

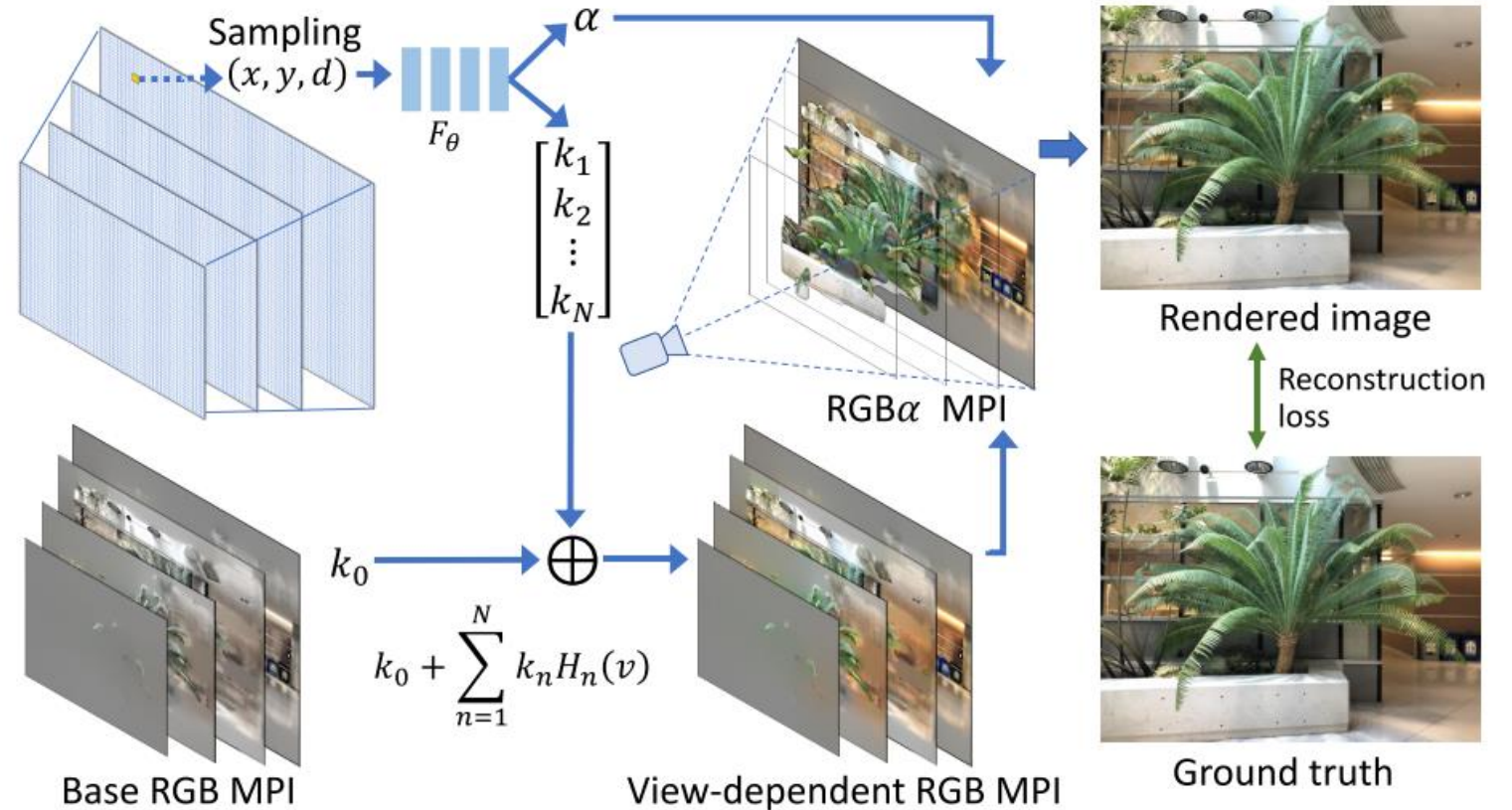
NeX: Real-time View Synthesis with Neural Basis Expansion



- ▶ MPI + view-dependent colour encoding
- ▶ High quality reproduction of the view-dependent effects
 - ▶ Specular reflections
 - ▶ Diffraction
 - ▶ ...

NeX: Real-time View Synthesis with Neural Basis Expansion

- ▶ The colour is encoded as a linear combination of the basis functions
- ▶ The basis functions are trainable



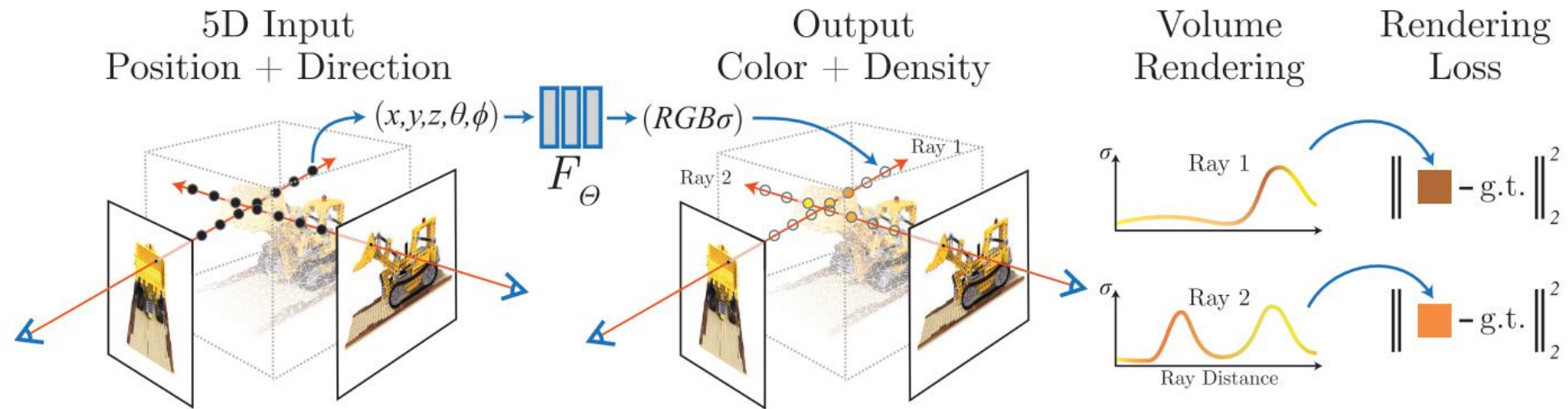
NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis

- ▶ Models a volume rather than a set of discrete planes
- ▶ 360 or front facing
- ▶ Uses MLP to represent the colour and opacity



Mildenhall, Ben, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis," 405–21, 2020. https://doi.org/10.1007/978-3-030-58452-8_24.

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis



Pixel
colour

(x, y, z) coordinates
along the ray

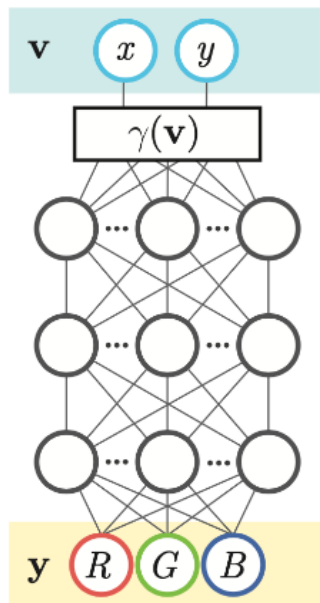
$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t) \sigma(\mathbf{r}(t)) \mathbf{c}(\mathbf{r}(t), \mathbf{d}) dt, \text{ where } T(t) = \exp\left(-\int_{t_n}^t \sigma(\mathbf{r}(s)) ds\right)$$

Computed as a
(differentiable) stratified
sampling

Colour at (x, y, z) in the
direction $\mathbf{d}$ (stored in an
MLP)

“opacity” or probability of
ray terminating (stored in
an MPL)

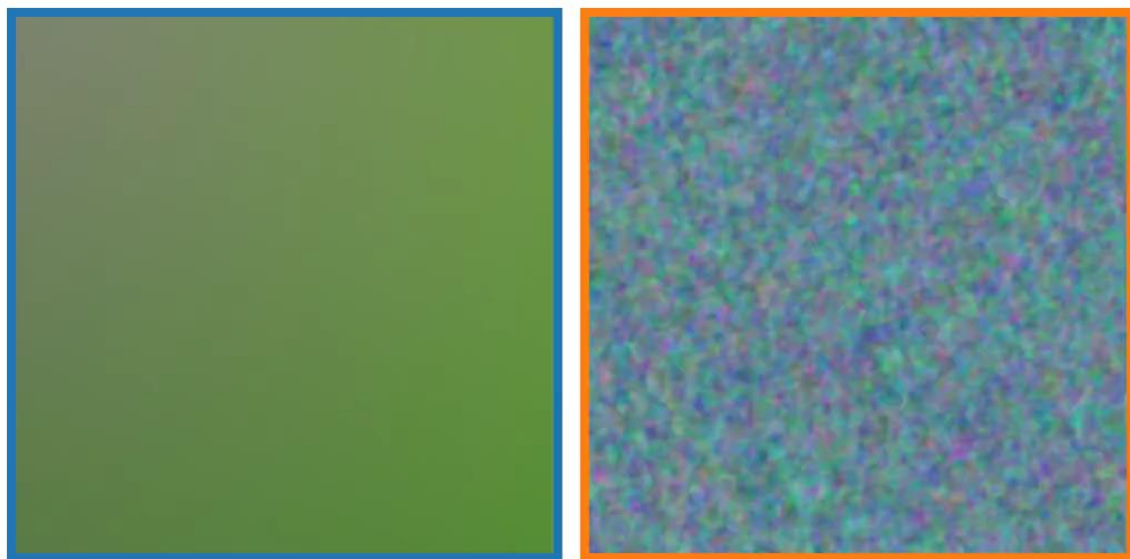
Positional encoding



$$\mathbf{y} = f(\gamma(\mathbf{p}); \mathbf{w})$$

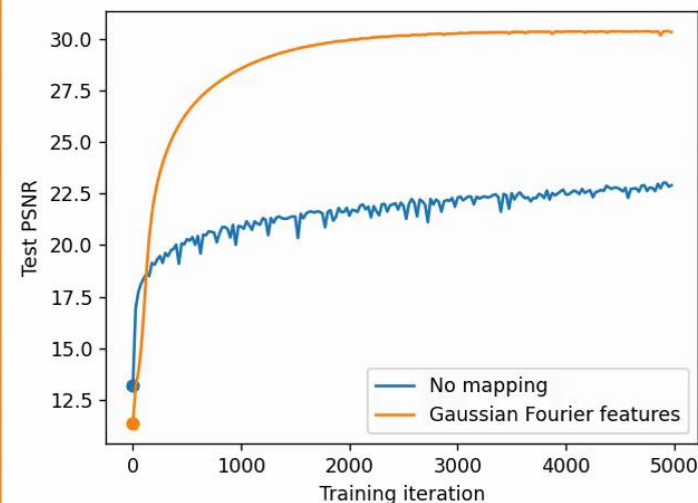
$$\gamma(\mathbf{p}) = (\sin(2^0 \pi p), \cos(2^0 \pi p), \dots, \sin(2^{L-1} \pi p), \cos(2^{L-1} \pi p))$$

- ▶ Encoding coordinates as the Fourier “features” allows MPL to learn high frequencies
- ▶ Works with other basis functions



No positional encoding

With positional encoding



Implicit (neural) (volumetric/n-dim) representations

- ▶ Neural signed distance function

- ▶ A function that stores a distance to a surface

- ▶ $d = f(x, y, z; \phi)$

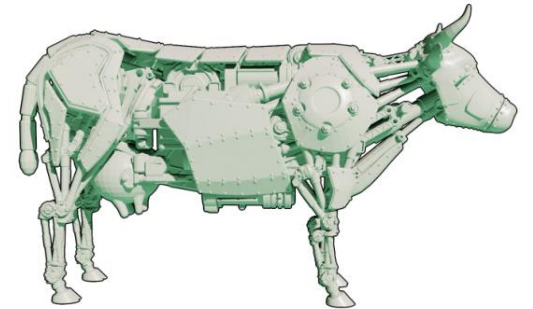
- ▶ Neural radiance caching

- ▶ Predict colour from feature buffers independently for each pixel

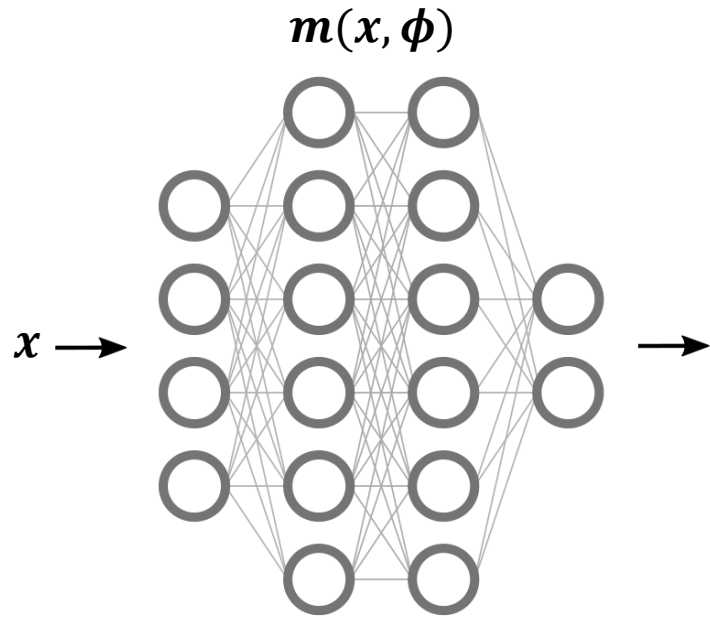


- ▶ Learning a giga-pixel image

- ▶ $RGB = f(x, y; \phi)$

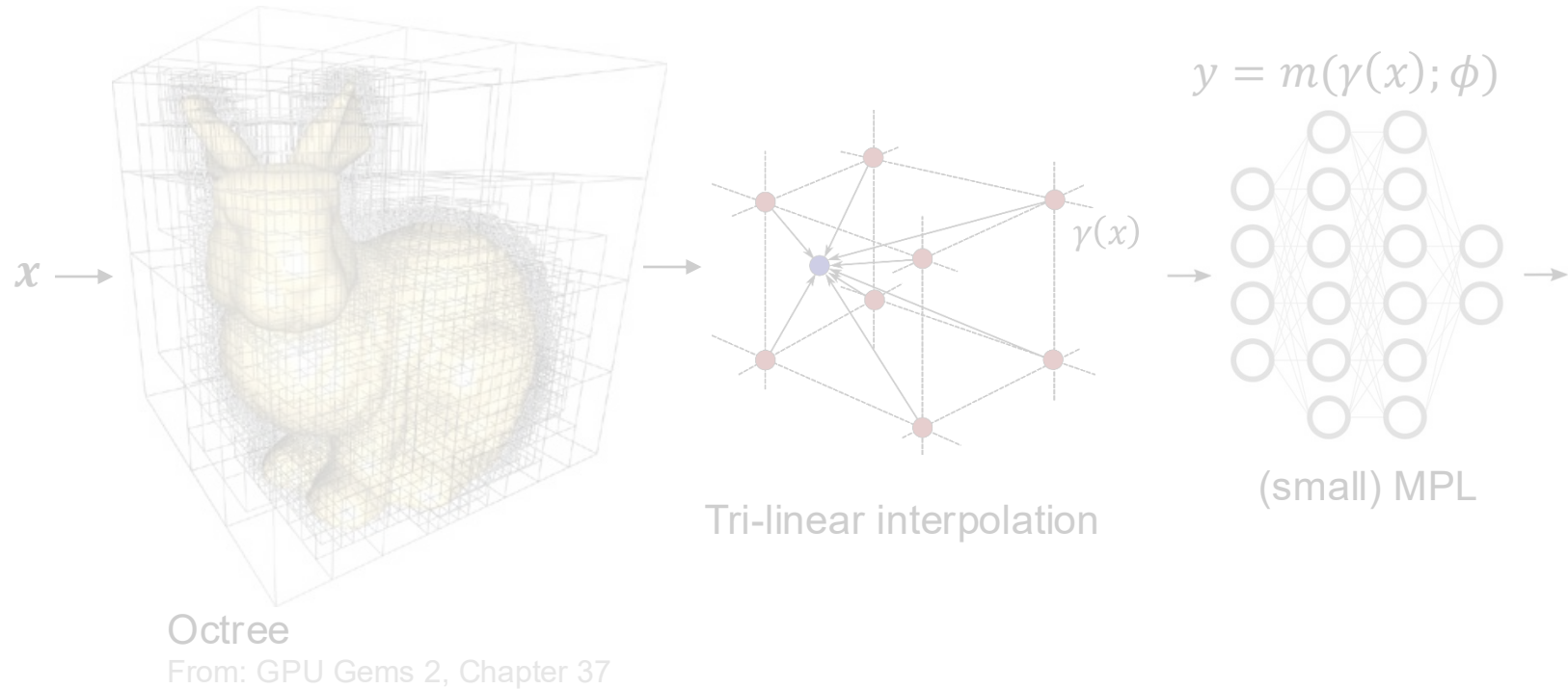


Reducing the cost of the MLP

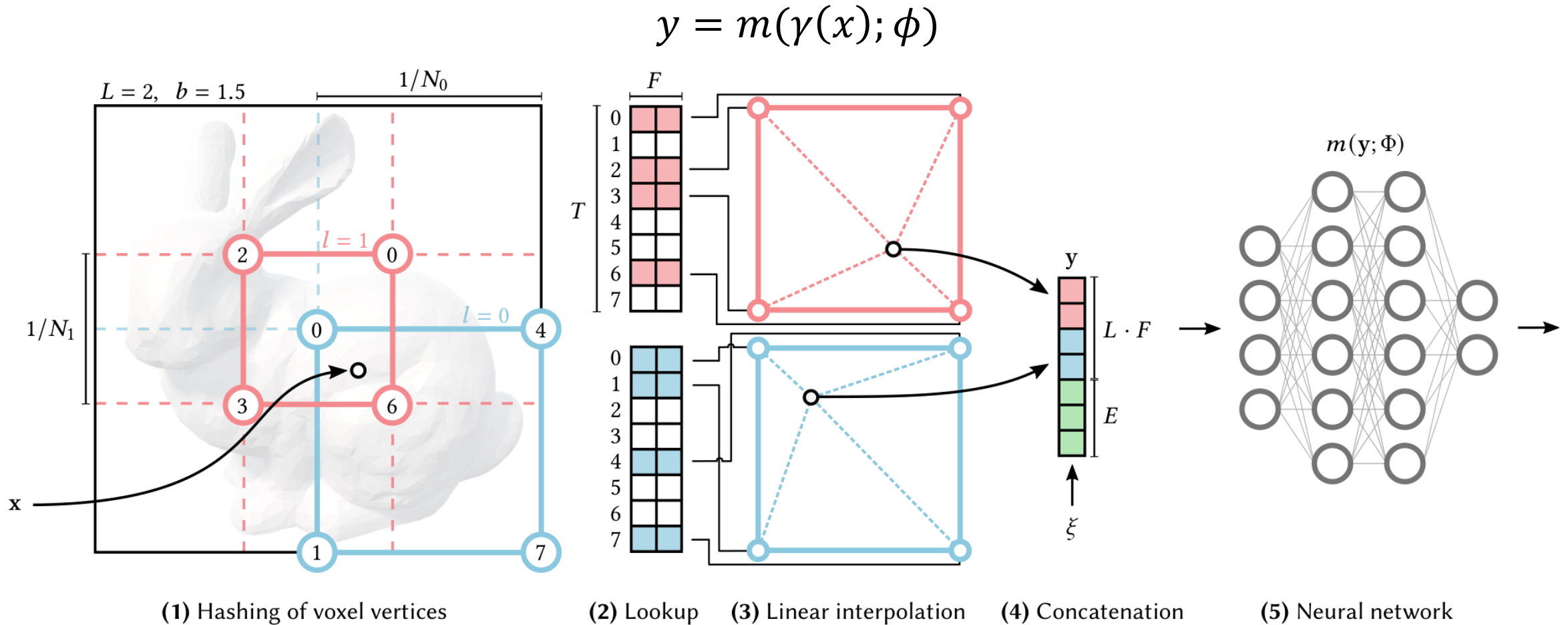


Given input coordinates x , only a small portion of the network activations will contribute to the output. This is inefficient.

Solution: encode a part of the information in a spatial data structure that can be directly queried, such as a (sparse) voxel grid or octree.

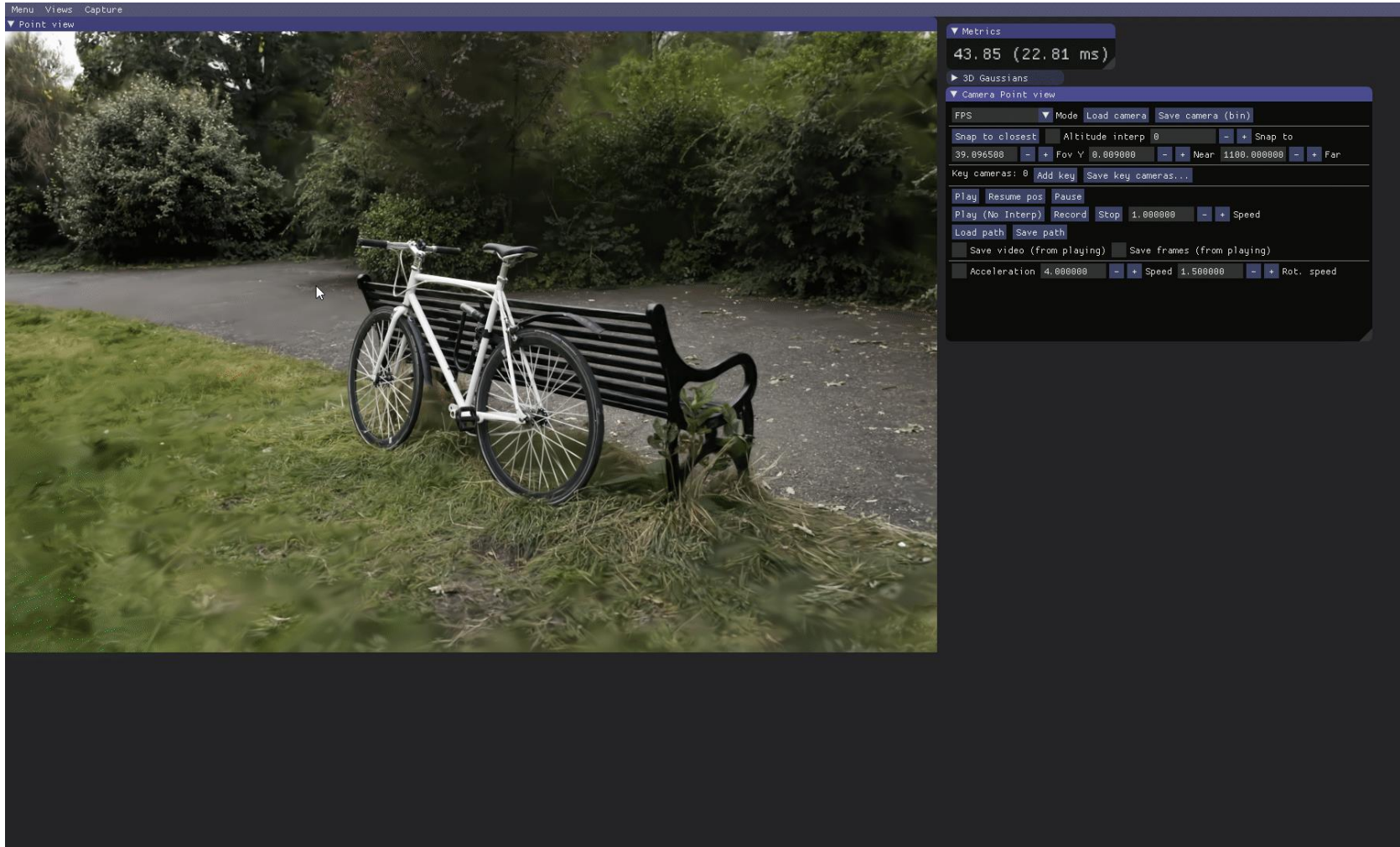


Instant neural graphics primitives with a multiresolution hash encoding



Müller, Thomas, Alex Evans, Christoph Schied, and Alexander Keller. "Instant Neural Graphics Primitives with a Multiresolution Hash Encoding." *ACM Transactions on Graphics* 41, no. 4 (July 22, 2022): 1–15. <https://doi.org/10.1145/3528223.3530127>.

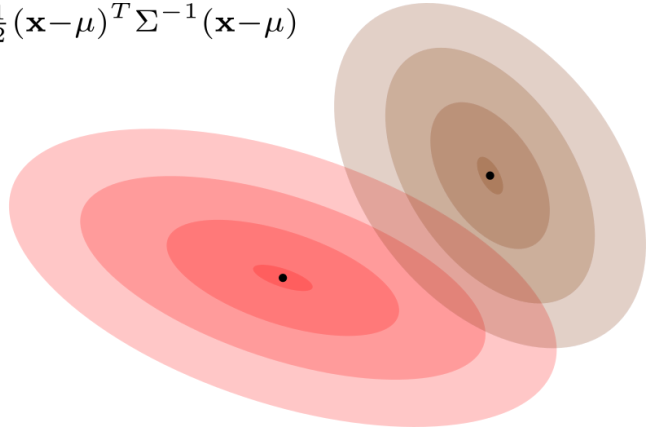
Gaussian Splatting (3DGS)



- ▶ Represent the scene as a cloud of semi-transparent 3D Gaussians
 - ▶ Render using point cloud rendering technique – splatting
 - ▶ Much faster rendering and training
 - ▶ Clever optimization
 - ☐ Tiled rendering
 - ☐ Renders only a fraction of Gaussians in each tile
 - ▶ But requires several millions of Gaussians

Gaussian Splatting (3DGS)

$$G(\mathbf{x}) = e^{\frac{1}{2}(\mathbf{x}-\mu)^T \Sigma^{-1}(\mathbf{x}-\mu)}$$

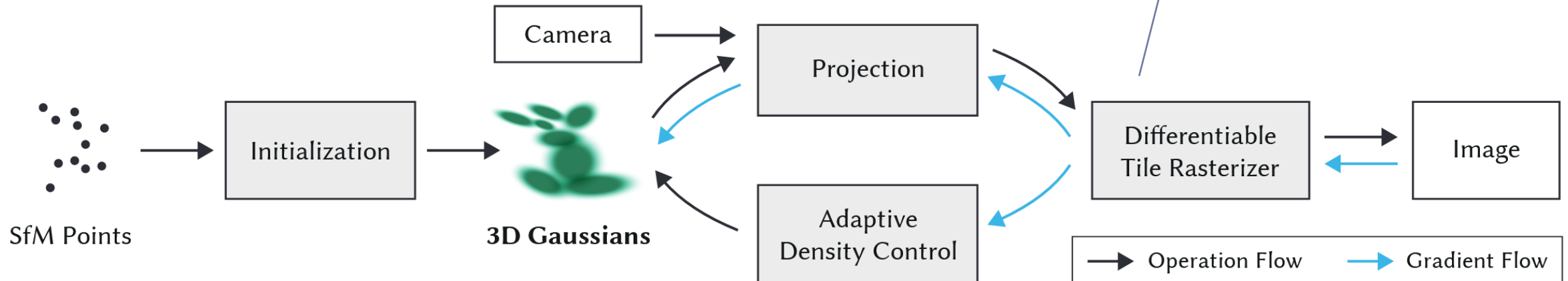


Each Gaussian is defined by

- ▶ Mean $\mu \in \mathbb{R}^3$
- ▶ Covariance matrix $\Sigma \in \mathbb{R}^{3 \times 3}$
- ▶ Opacity $o \in \mathbb{R}$
- ▶ Directional colour (as Spherical Harmonics) $c \in \mathbb{R}^{16 \times 3}$ or $c \in \mathbb{R}^{9 \times 3}$

Rendering

- ▶ Project 3D→2D
- ▶ Sort Gaussians according to depths
- ▶ For each pixel
 - Accumulate colour using the volumetric rendering equation (as in NeRF)



Spherical Harmonics (SH)

- ▶ Othonormal basis defined on a sphere
 - ▶ Think: Fourier basis on a sphere
- ▶ Directional functions (e.g., view-dependent colour) can be approximated using SH basis

$$f(r, \theta, \varphi) = \sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} f_{\ell}^m r^{\ell} Y_{\ell}^m(\theta, \varphi)$$

