UNIVERSITY OF CAMBRIDGE
COMPUTER LABORATORY

RAINBOW GRAPHICS GROUP

Advanced Graphics & Image Processing

Image-based rendering and Light fields

Part 1/4 – context, definition and technology

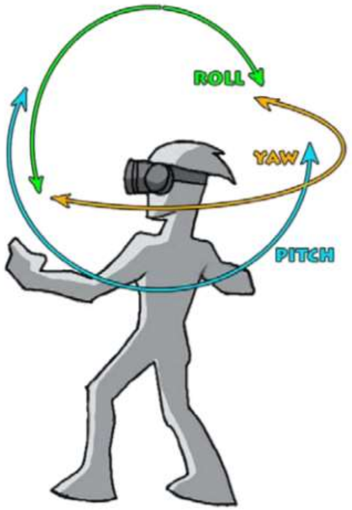
Rafał Mantiuk

Computer Laboratory, University of Cambridge

Motivation: 3DoF vs 6DoF in VR

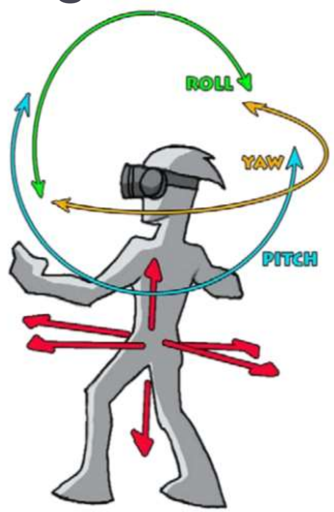
3DoF

- Tracking with inexpensive Inertial Measurements Units
- Content:
  - Geometry-based graphics
  - Omnidirectional stereo video
    - May induce cyber-sickness due due to the lack of motion depth cues



6DoF

- Requires internal (inside-out) or external tracking
- Content:
  - Geometry-based graphics
  - Point-cloud rendering
  - Image-based rendering
    - View interpolation
    - **Light fields**
    - …



Source for the images: https://www.qualcomm.com/media/documents/files/on-device-motion-tracking-for-immersive-vr.pdf

3D computer graphics

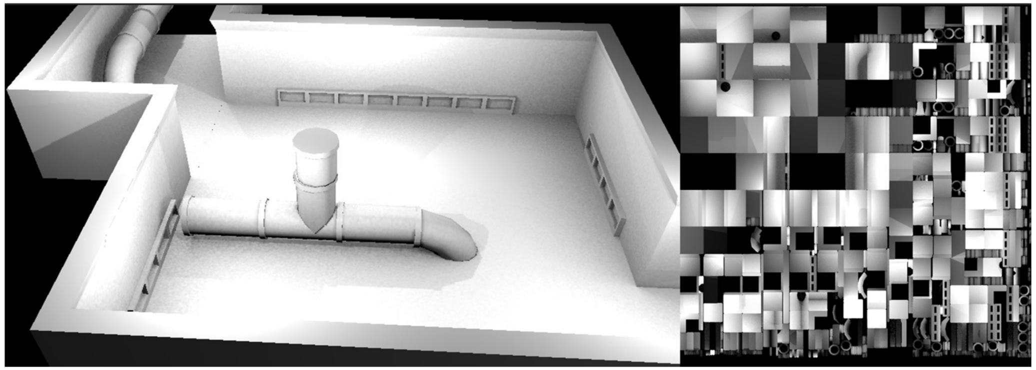
- We need:
  - Geometry + materials + textures
  - Lights
  - Camera
- Full control of illumination, realistic material appearance
- Graphics assets are expensive to create
- Rendering is expensive
  - Shading tends to takes most of the computation

Cyberpunk 2077 (C) 2020 by CD Projekt RED

Baked / precomputed illumination

- We need:
  - Geometry + textures + (light maps)
  - Camera
- No need to scan/model materials
- Much faster rendering – simplified shading

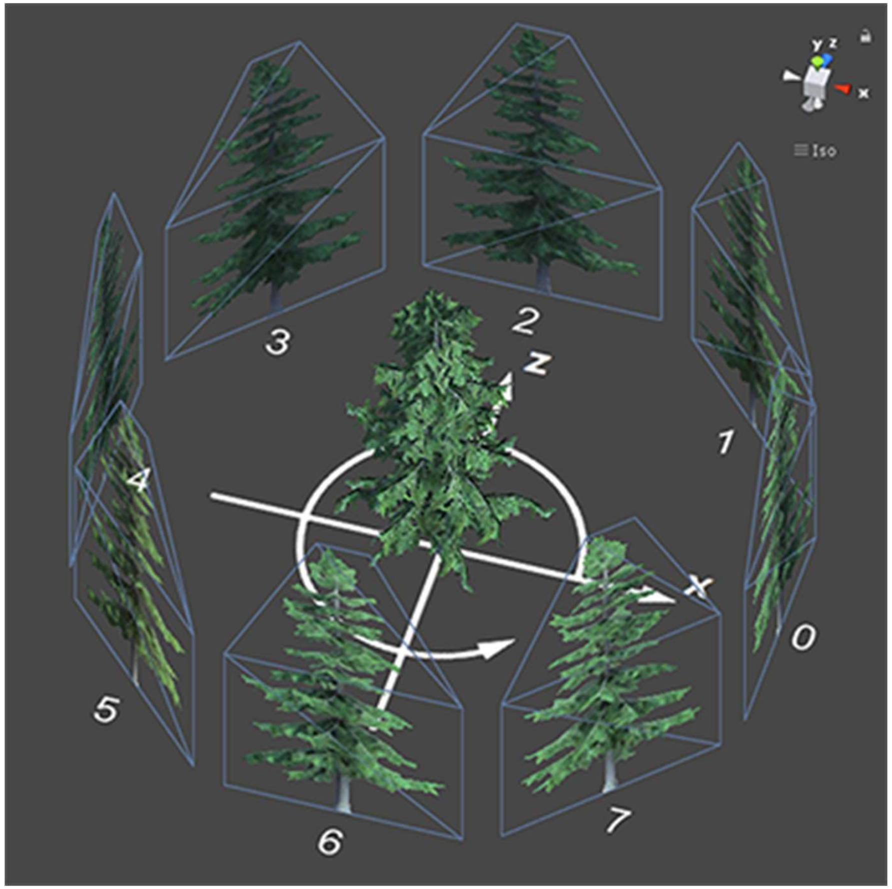
Google Earth



Precomputed light maps (from Wikipedia)

Billboards / Sprites

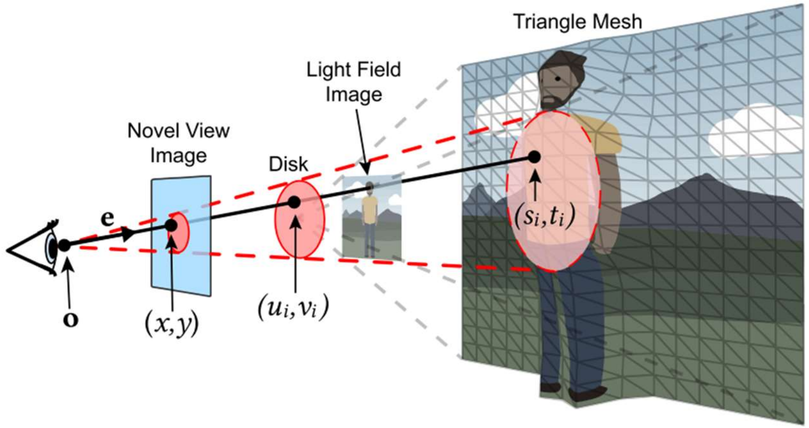
- We need:
  - Simplified geometry + textures (with alpha)
  - Lights
  - Camera
- Much faster to render than objects with 1000s of triangles
- Used for distant objects
  - or a small rendering budget
- Can be pre-computed from complex geometry



A tree rendered from a set of billboards
From: https://docs.unity3d.com/ScriptReference/BillboardAsset.html

Light fields + depth

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



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

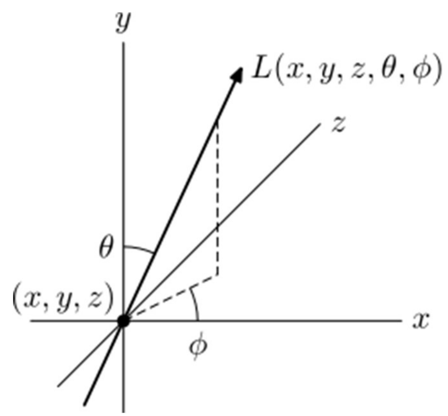
Demo: https://augmentedperception.github.io/welcome-to-lightfields/

Light fields

- We need:
  - Images of the scene
    - Or a microlens image
  - Camera
- As light fields +depth but
  - No geometry, no need for any 3D reconstruction
  - Photographs are rep-projected on the plane
  - Requires massive number of images for good quality

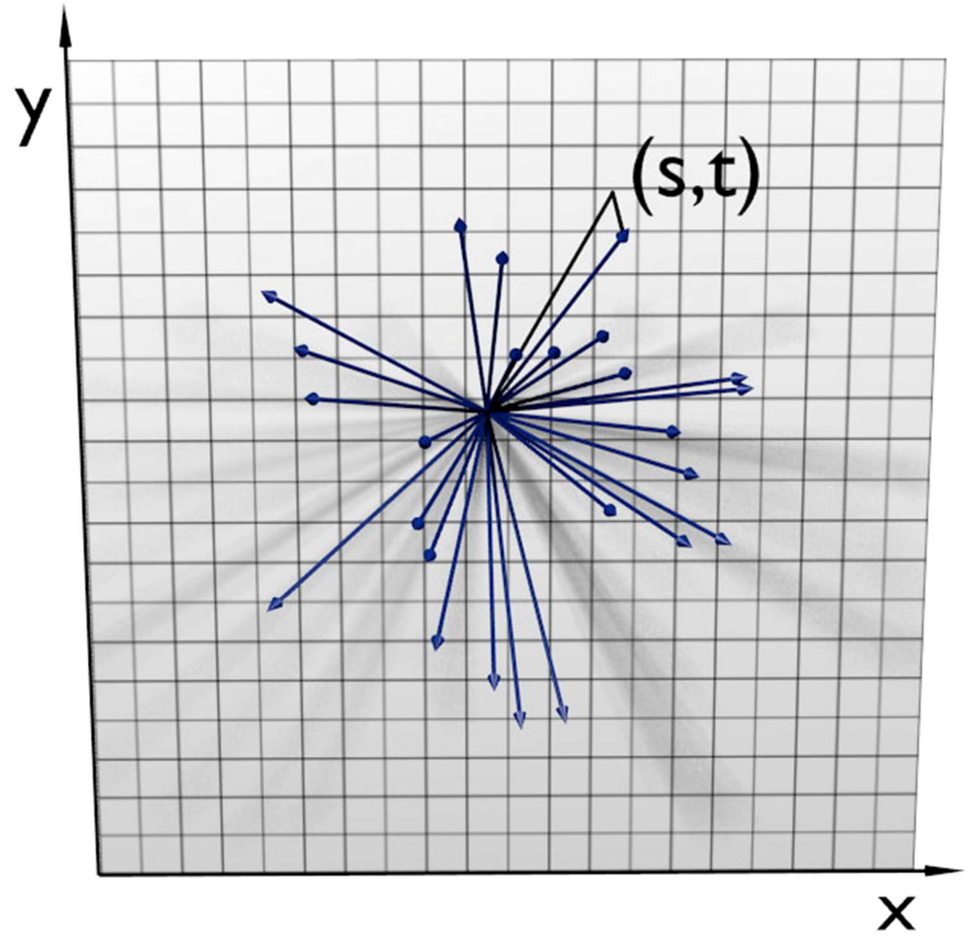
From a plenoptic function to a light field

- 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

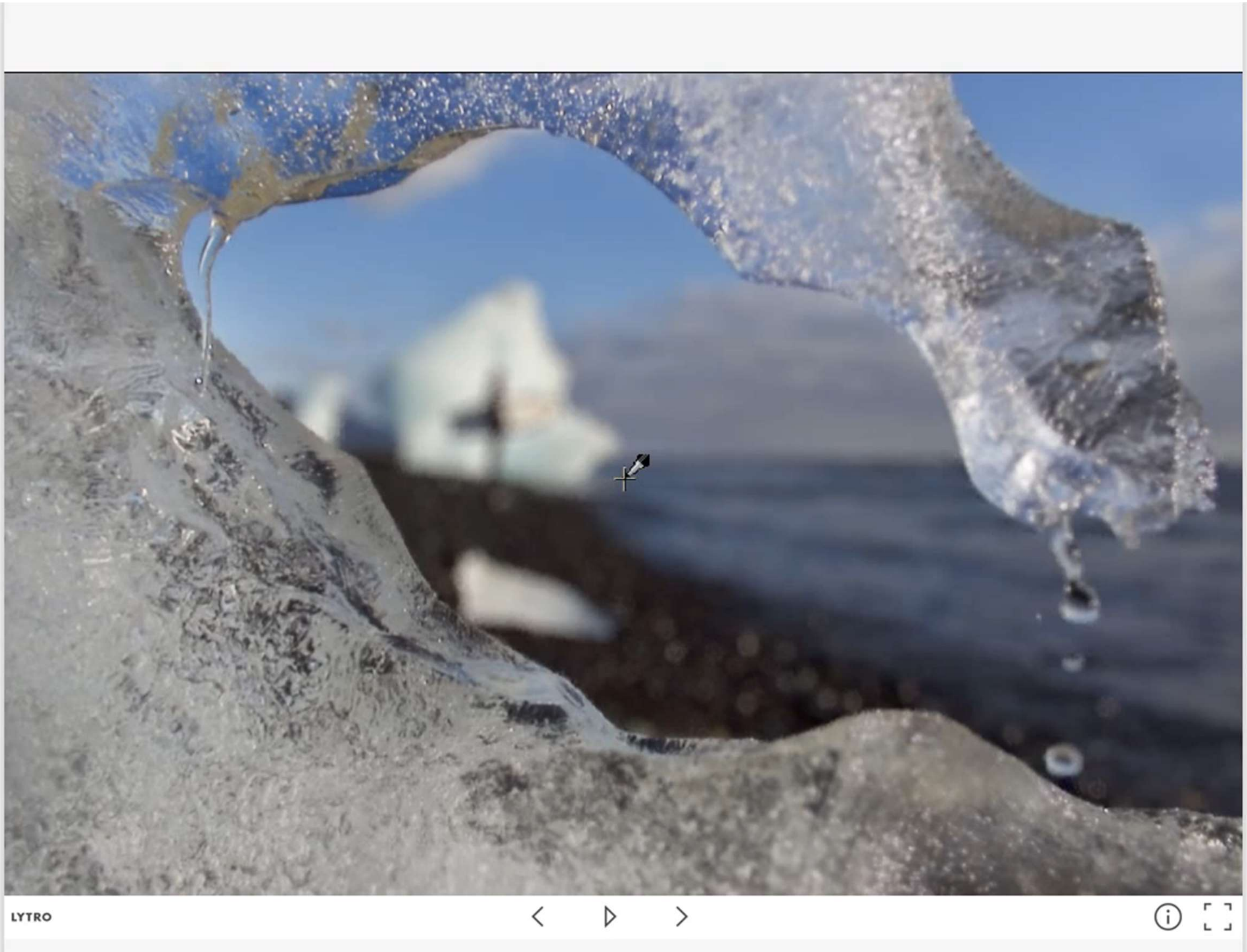


- But the number of dimensions can be reduced if
  - The camera stays outside the convex hull of the object
  - The light travels in uniform medium
  - Then, radiance L remains the same along the ray (until the ray hits an object)
  - This way we obtain a **4D light field** or **lumigraph**

Planar 4D light field



Refocusing and view point adjustment



Screen capture from http://www.lytro.com/

Depth estimation from light field

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

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

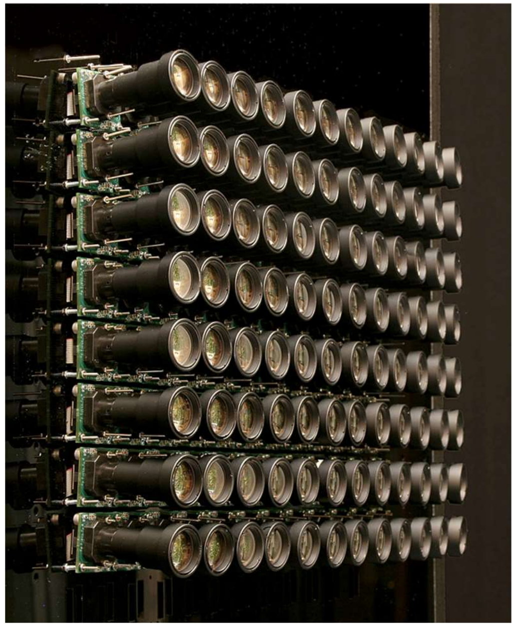
Two methods to capture light fields

Micro-lens array

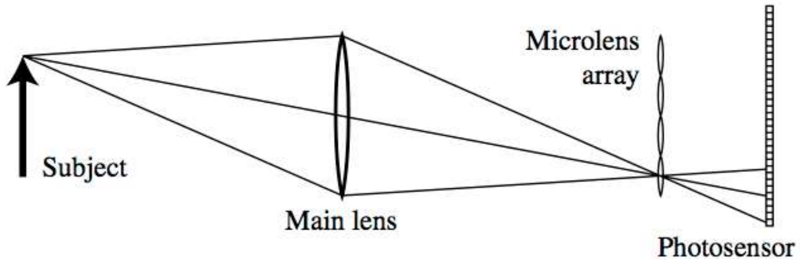
- Small baseline
- Good for digital refocusing
- Limited resolution

Camera array

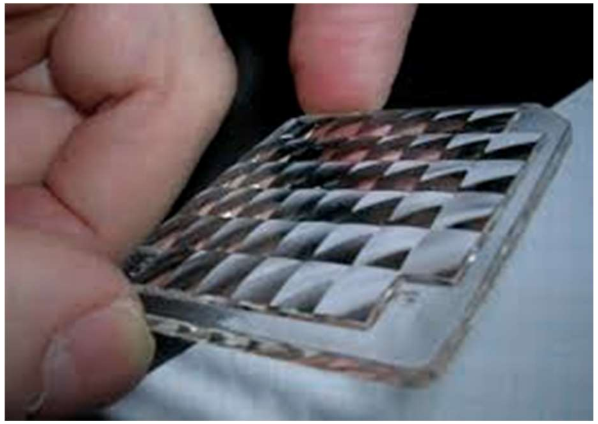
- Large baseline
- High resolution
- Rendering often requires approximate depth



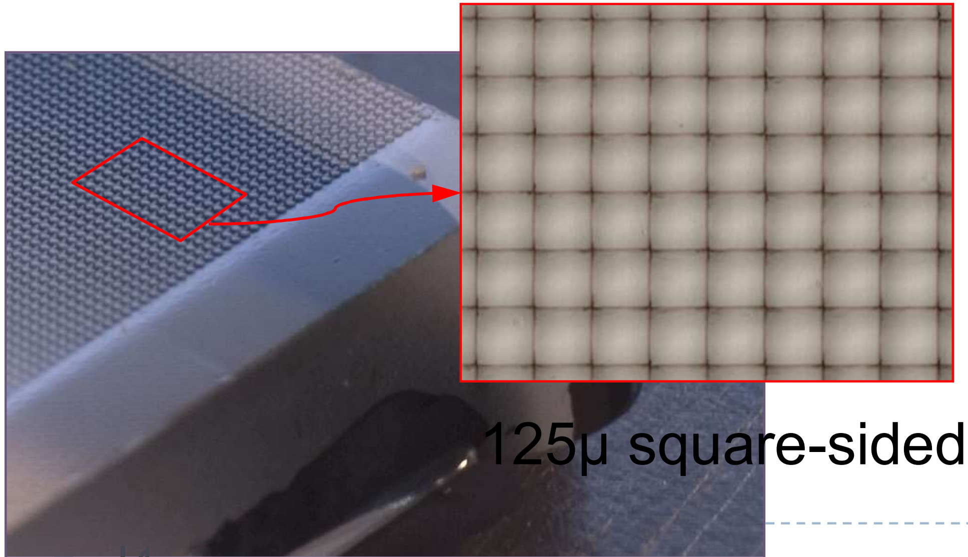
Light field image – with microlens array



Digital Refocusing using Light Field Camera



Lenslet array



125μ square-sided microlenses

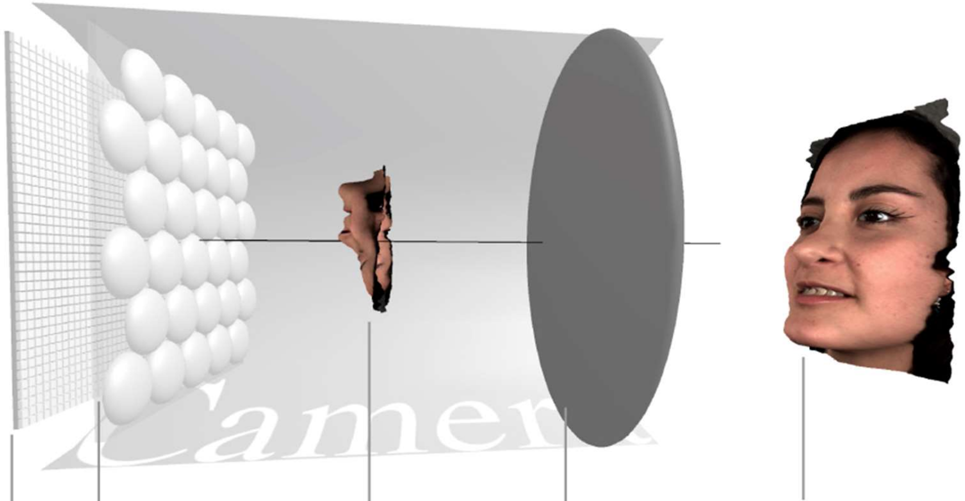
[Ng et al 2005]

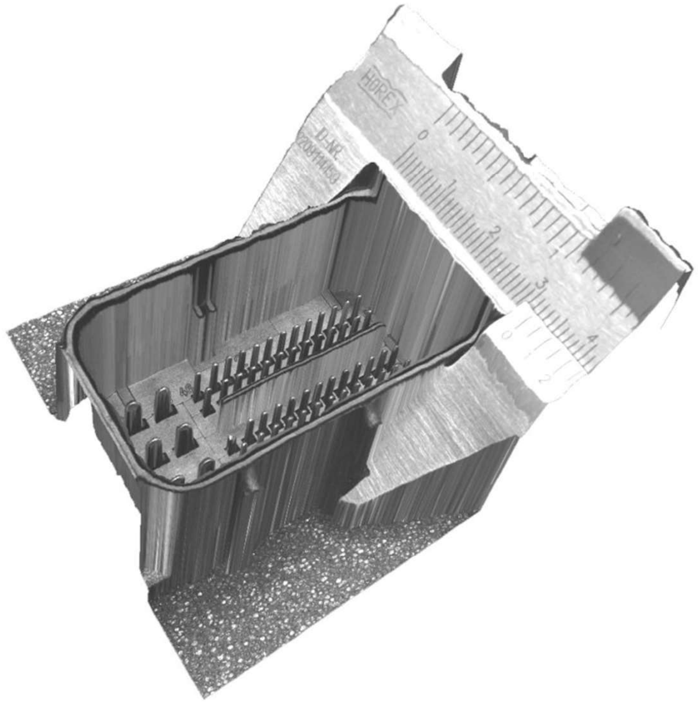
Lytro-cameras

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

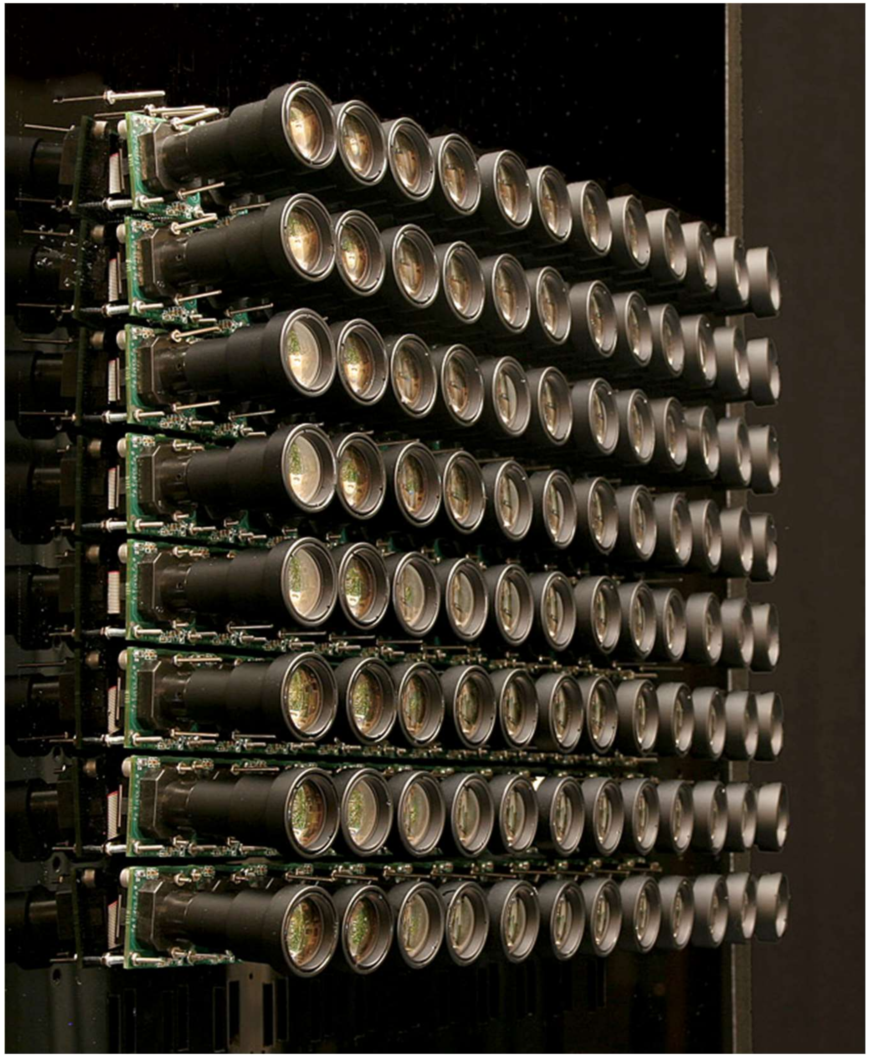
Raytrix camera

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



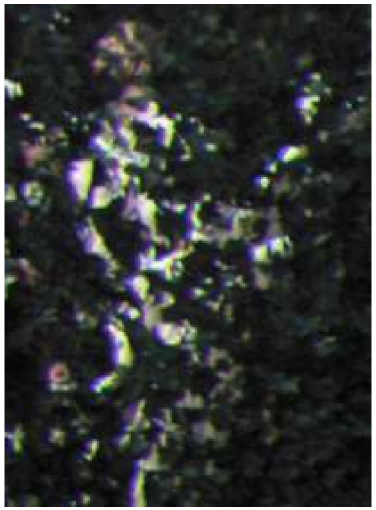


Stanford camera array



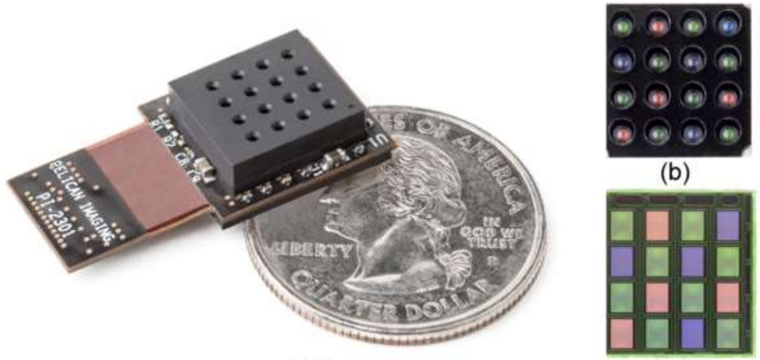
96 cameras

Application: Reconstruction of occluded surfaces

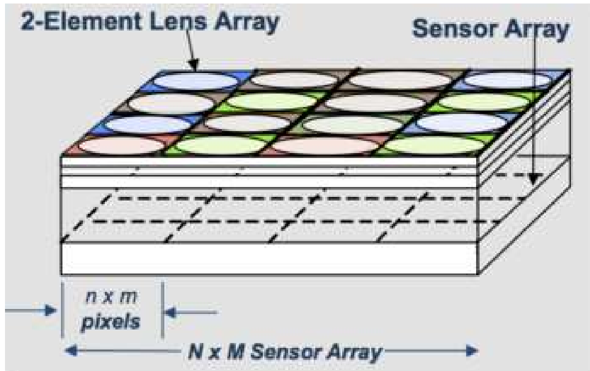


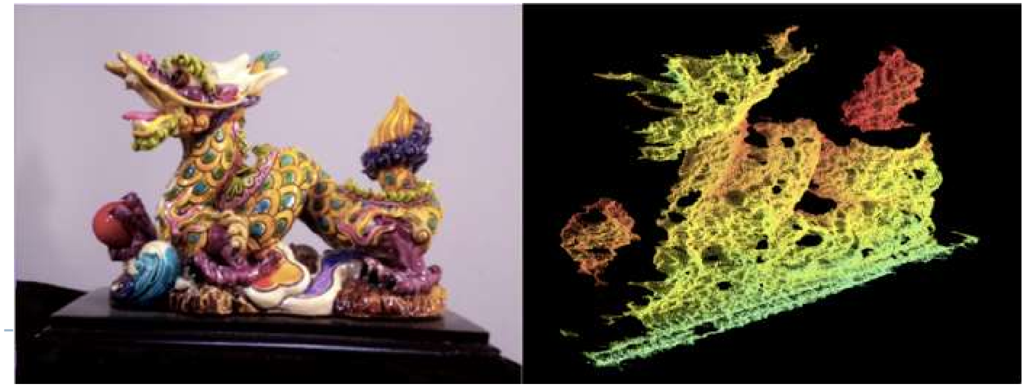
PiCam camera array module

- 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



(a) (b) (c)





UNIVERSITY OF CAMBRIDGE
COMPUTER LABORATORY

RAINBOW GRAPHICS GROUP

Advanced Graphics & Image Processing

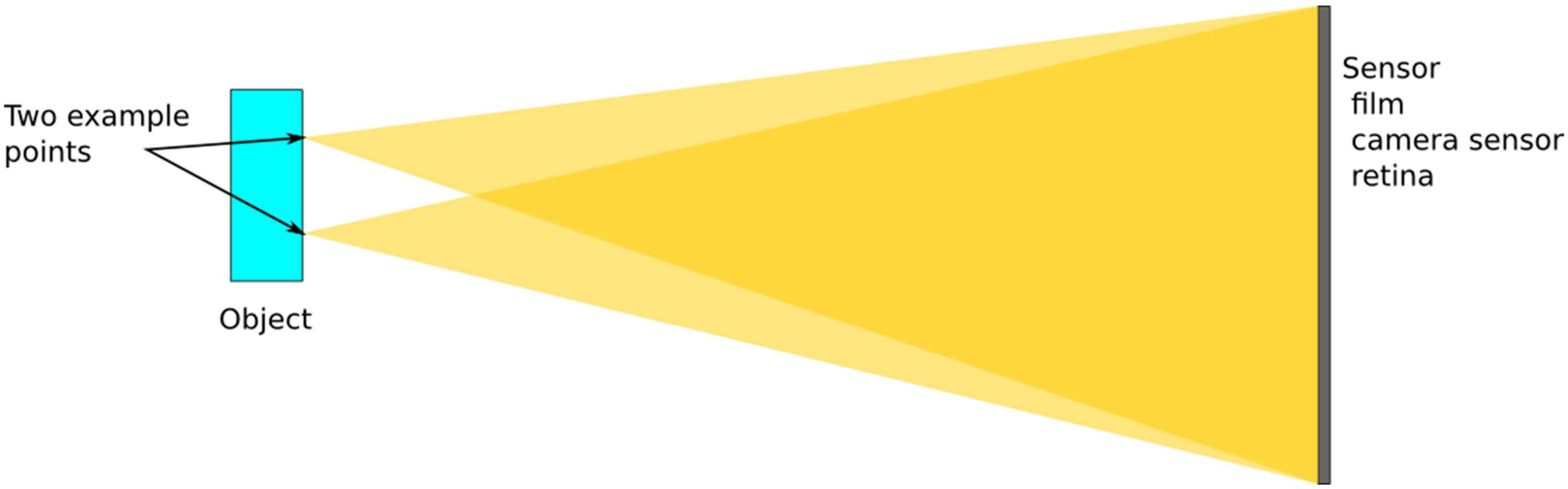
Light fields

Part 2/4 – imaging and lens

Rafał Mantiuk

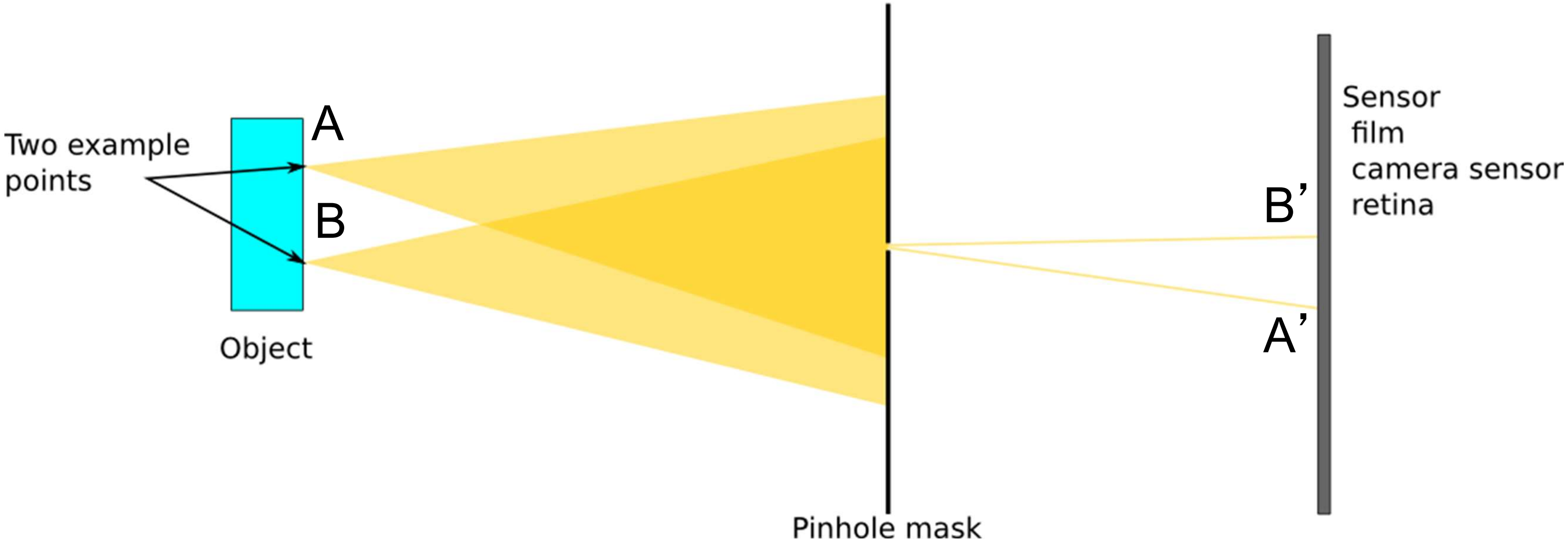
Computer Laboratory, University of Cambridge

Imaging – without lens



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

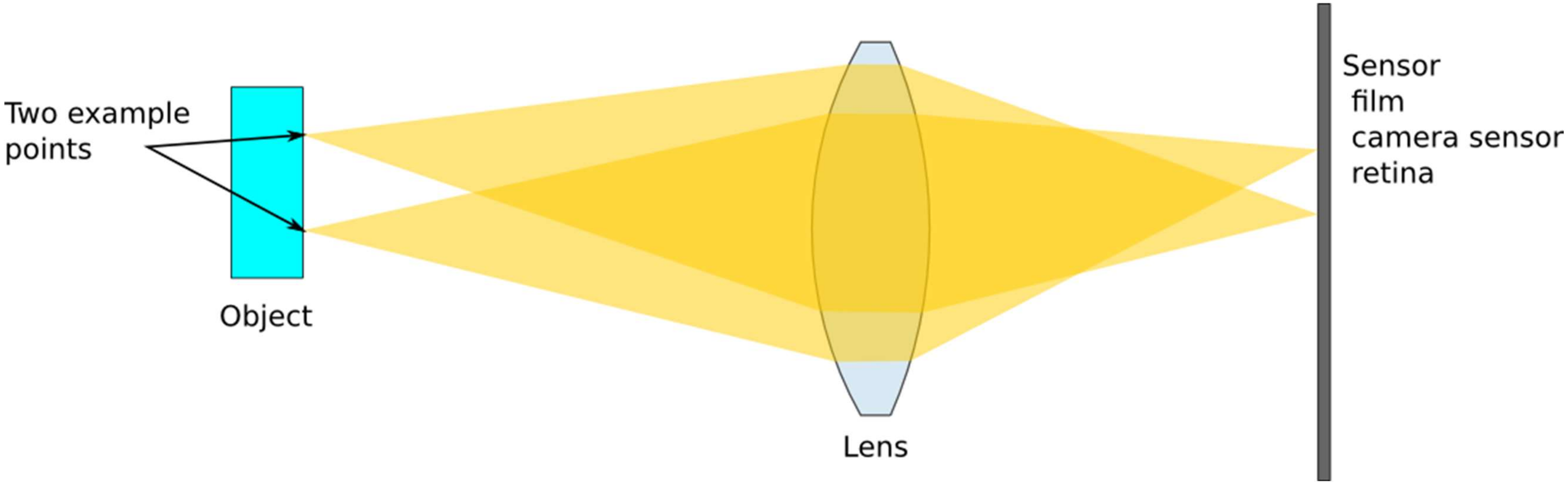
Imaging – pinhole camera



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.

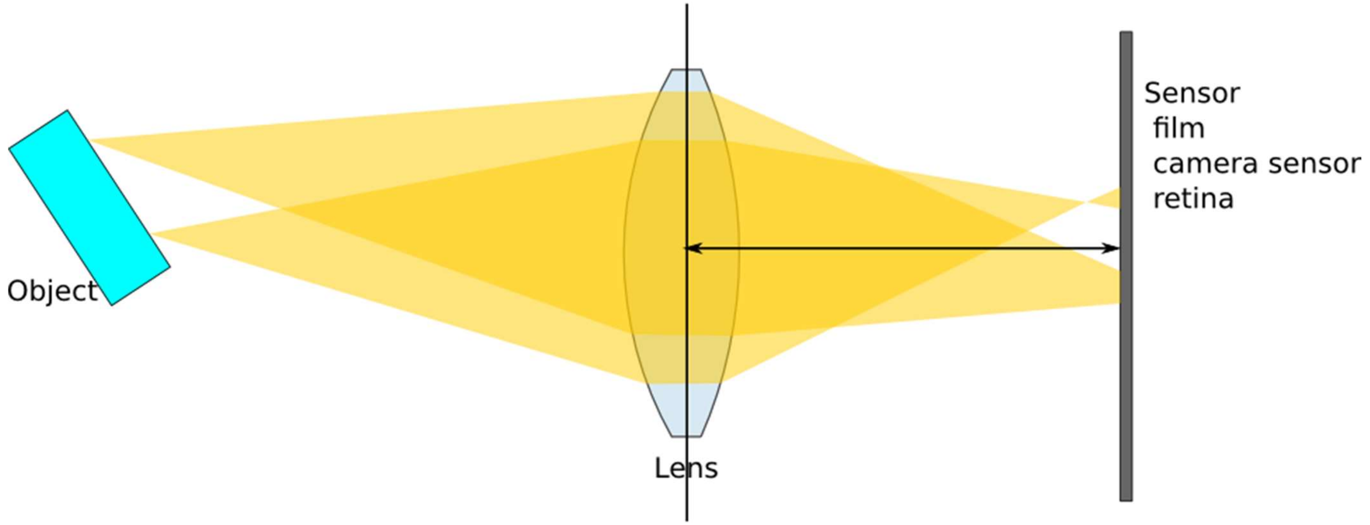
Imaging – lens



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

Much more light-efficient than the pinhole.

Imaging – lens

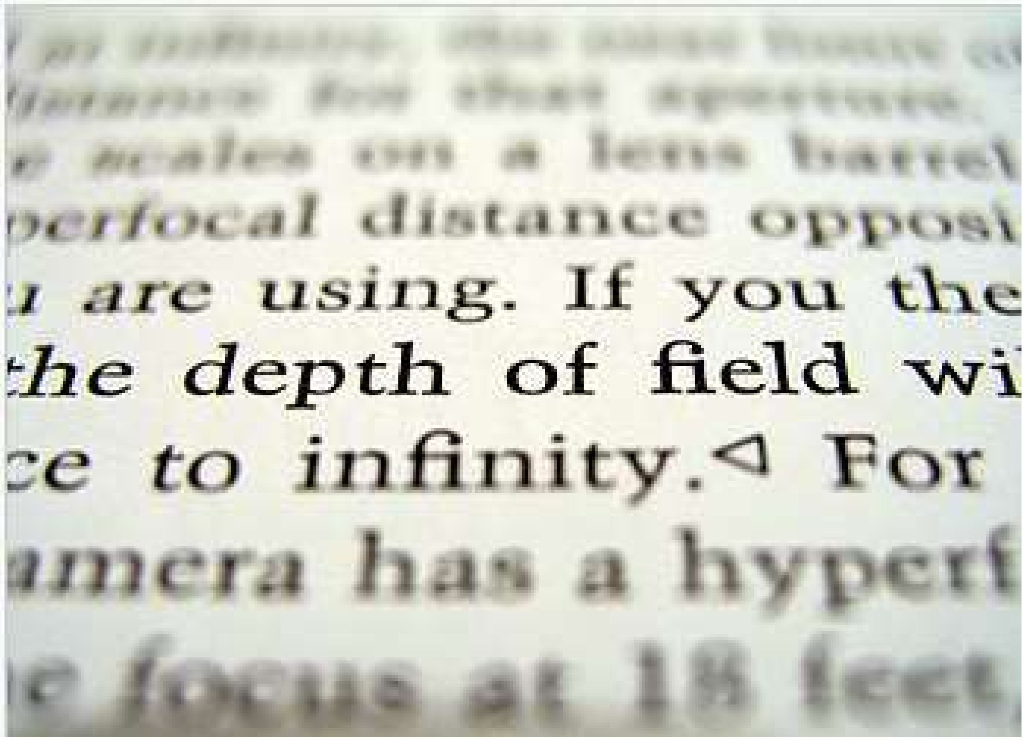


But it the light beams coming from different distances are not focused on the same plane.
These points will appear blurry in the resulting image.

Camera needs to move lens to focus an image on the sensor.

Depth of field

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

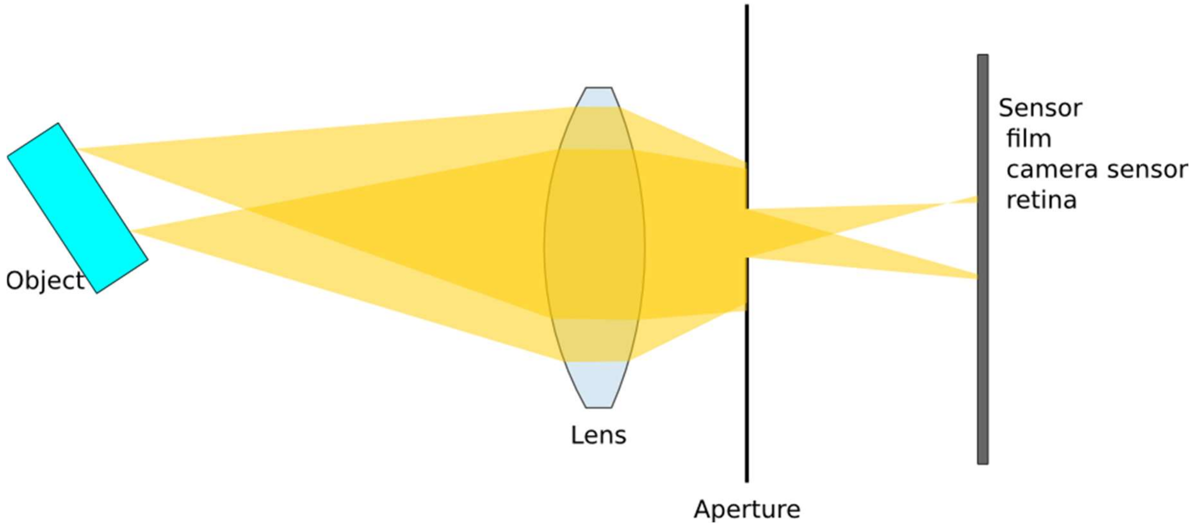


Defocus blur is often desirable

To separate the object of interest from background

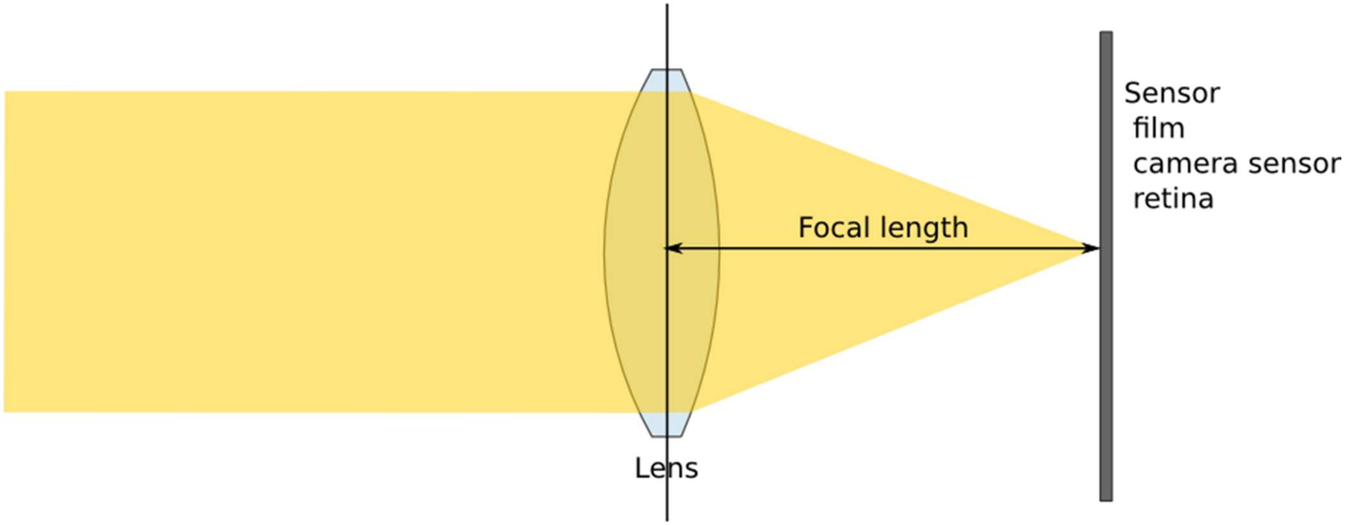
Defocus blur is a strong depth cue

Imaging – aperture



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

Imaging – lens



Focal length – length 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?

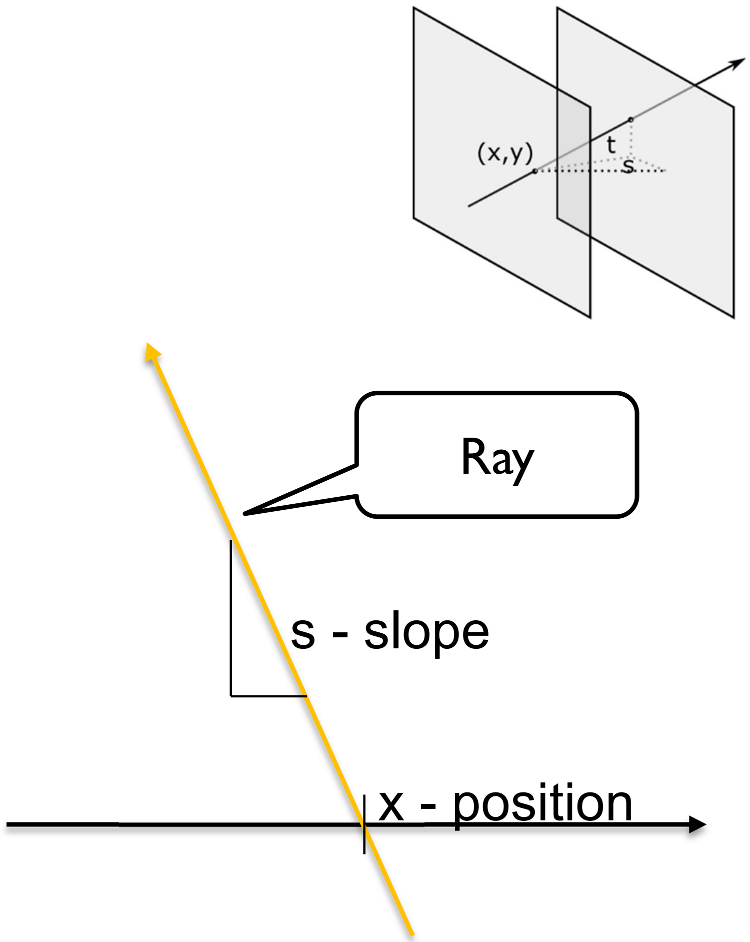
Advanced Graphics & Image Processing

Light fields

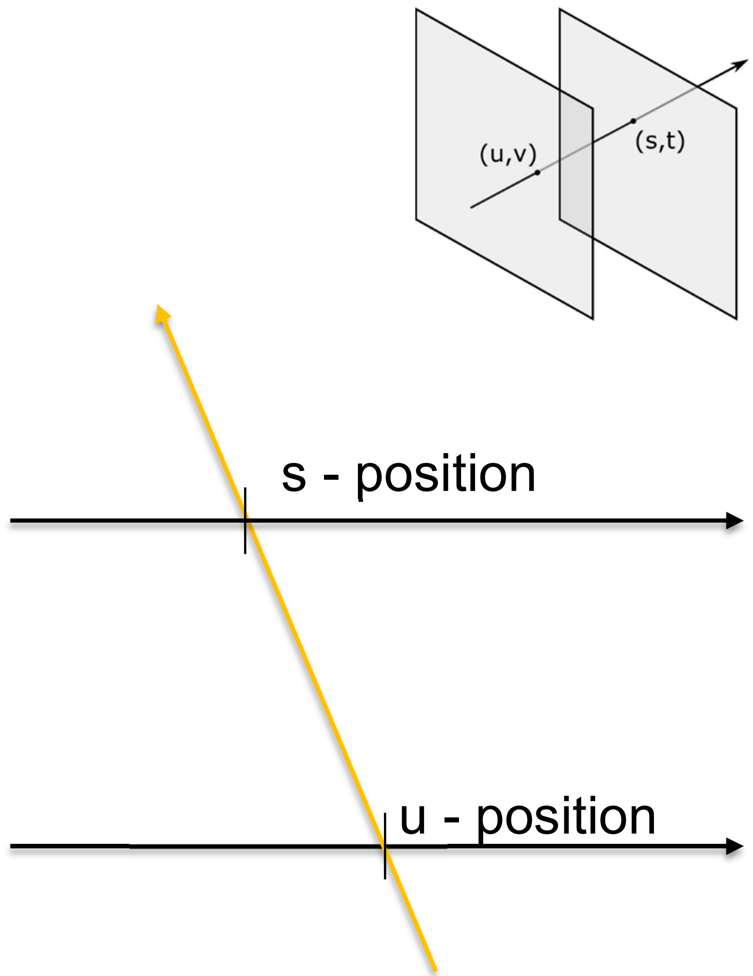
Part 3/4 – parametrization and an example

Rafał Mantiuk
Computer Laboratory, University of Cambridge

Light fields: two parametrisations (shown in 2D)

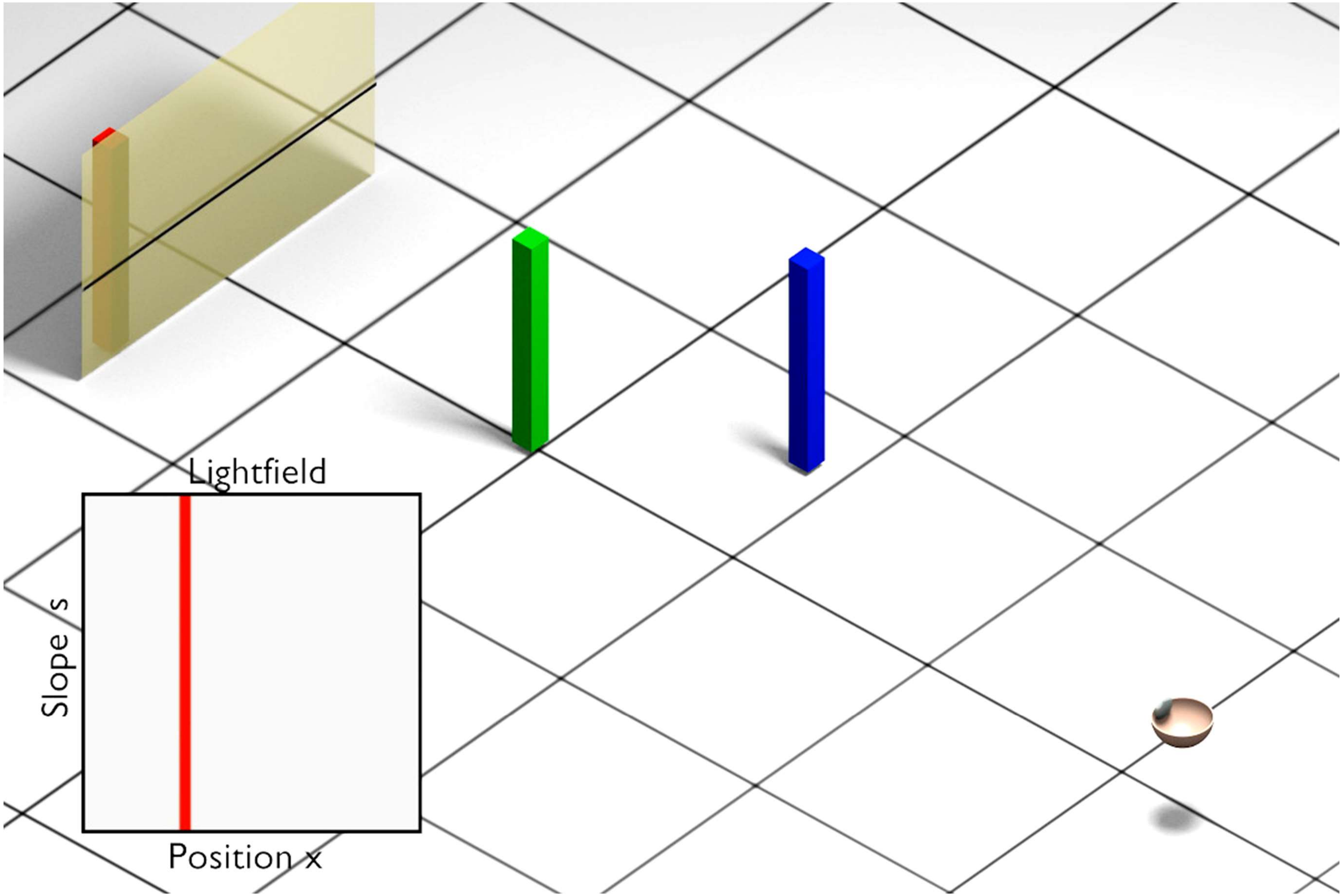


Position and slope
(slope - tangent of the angle)

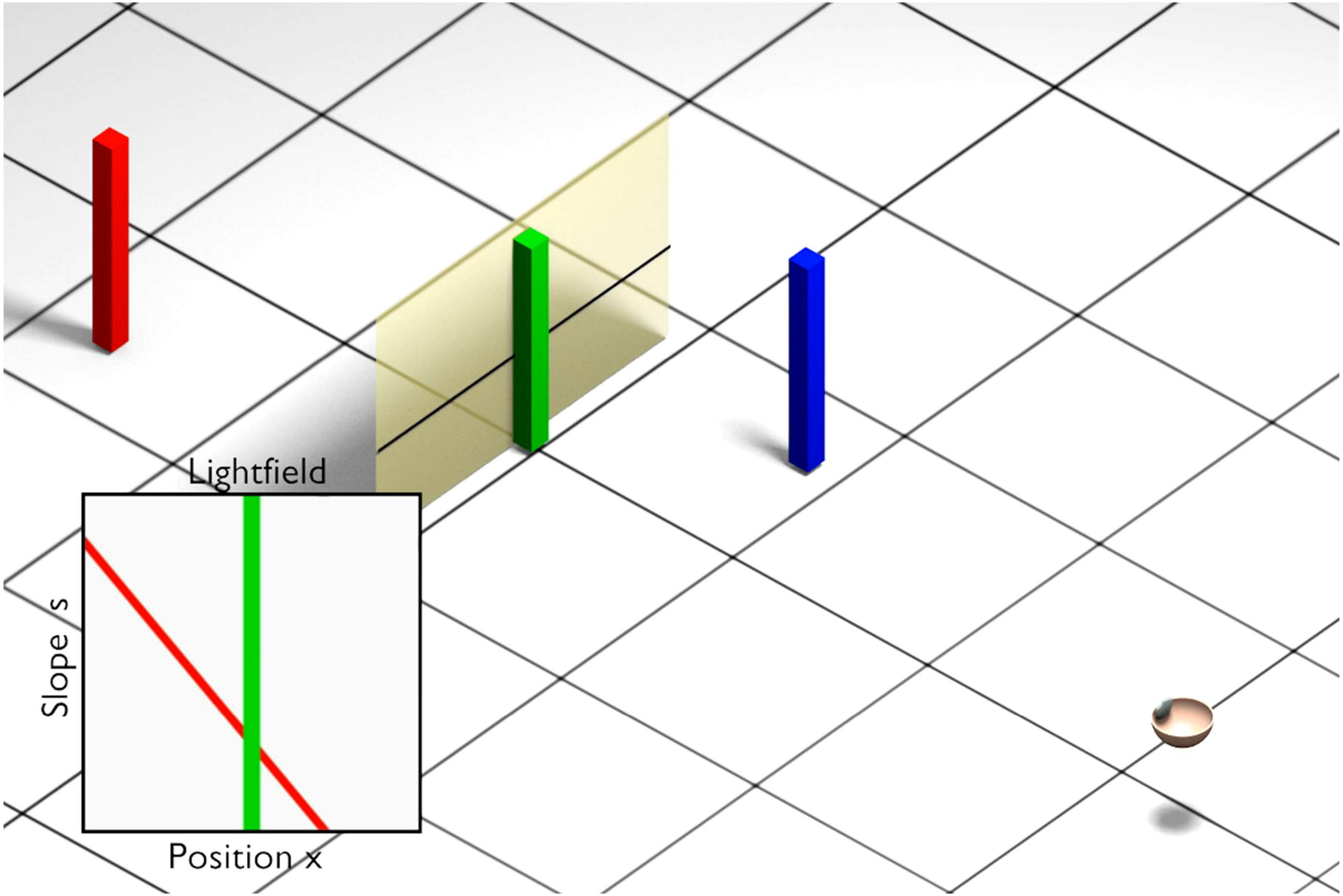


Two planes

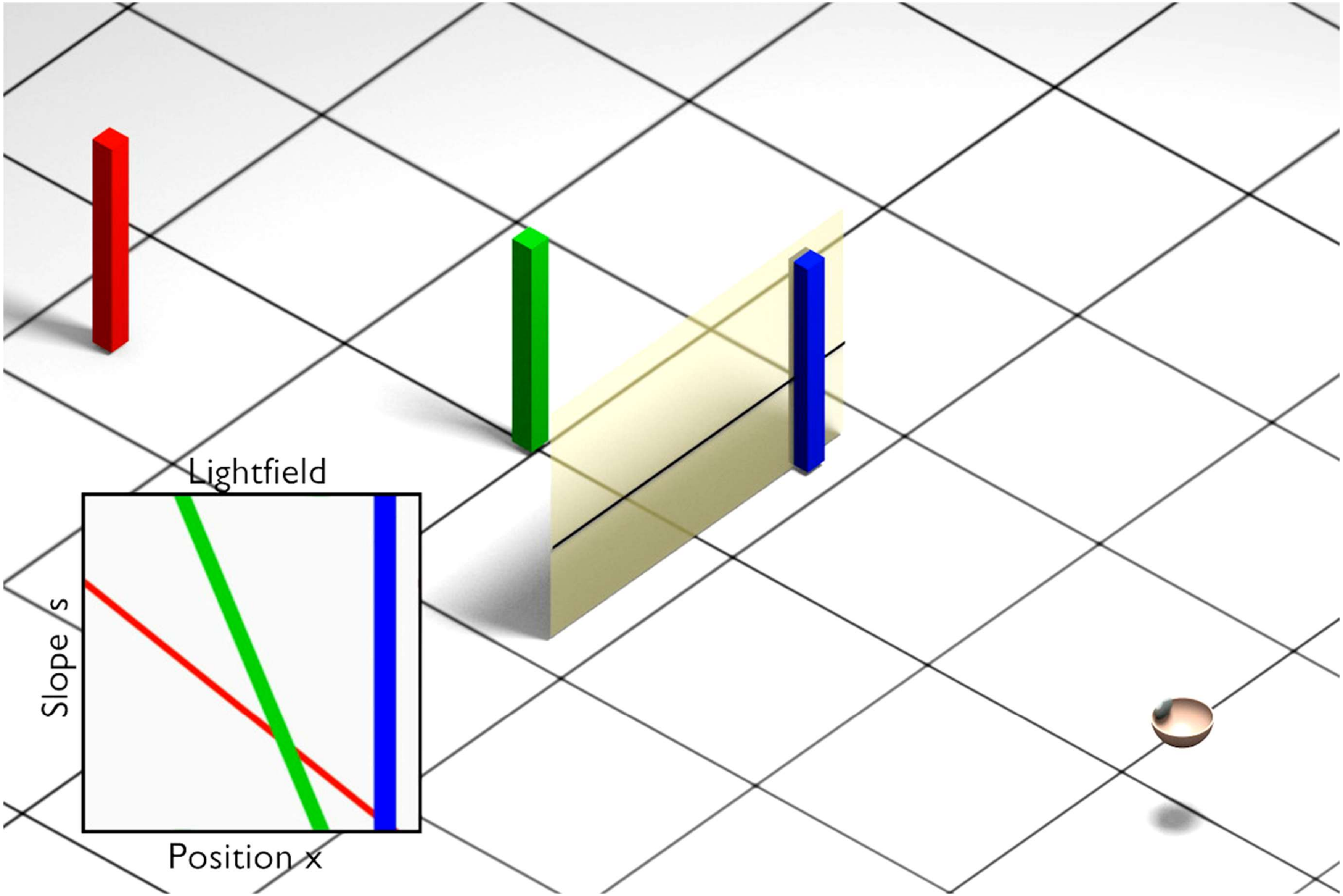
Lightfield - example



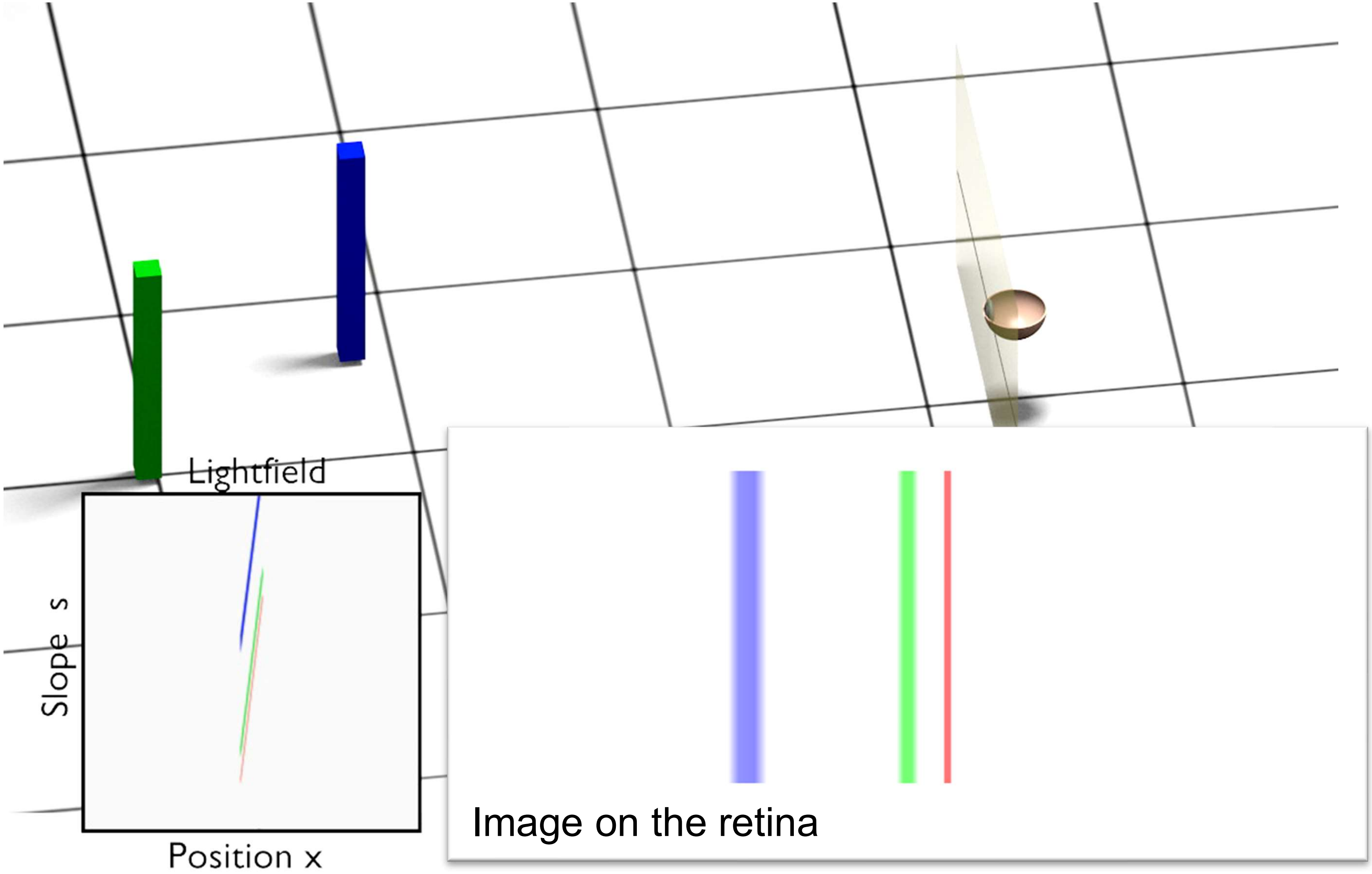
Lightfield - example



Lightfield - example



Lightfield - example



UNIVERSITY OF CAMBRIDGE
COMPUTER LABORATORY
RAINBOW GRAPHICS GROUP

Advanced Graphics & Image Processing

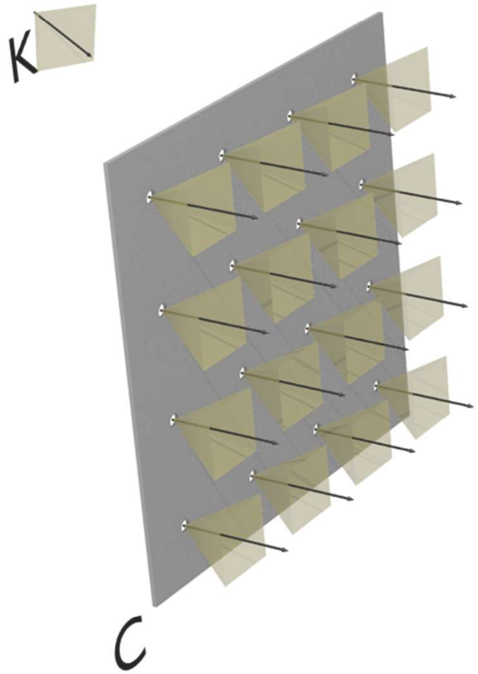
Light fields

Part 4/4 – light field rendering

Rafał Mantiuk

Computer Laboratory, University of Cambridge

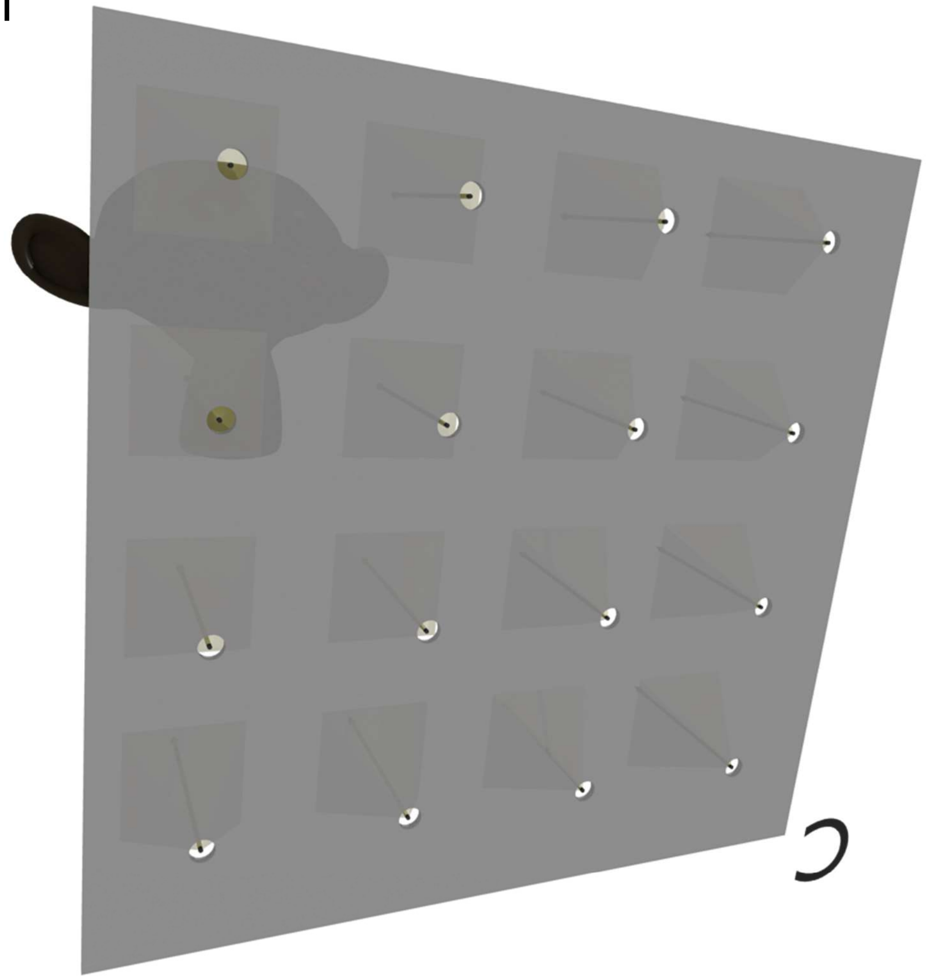
Light field rendering (1/3)



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 view point of camera K

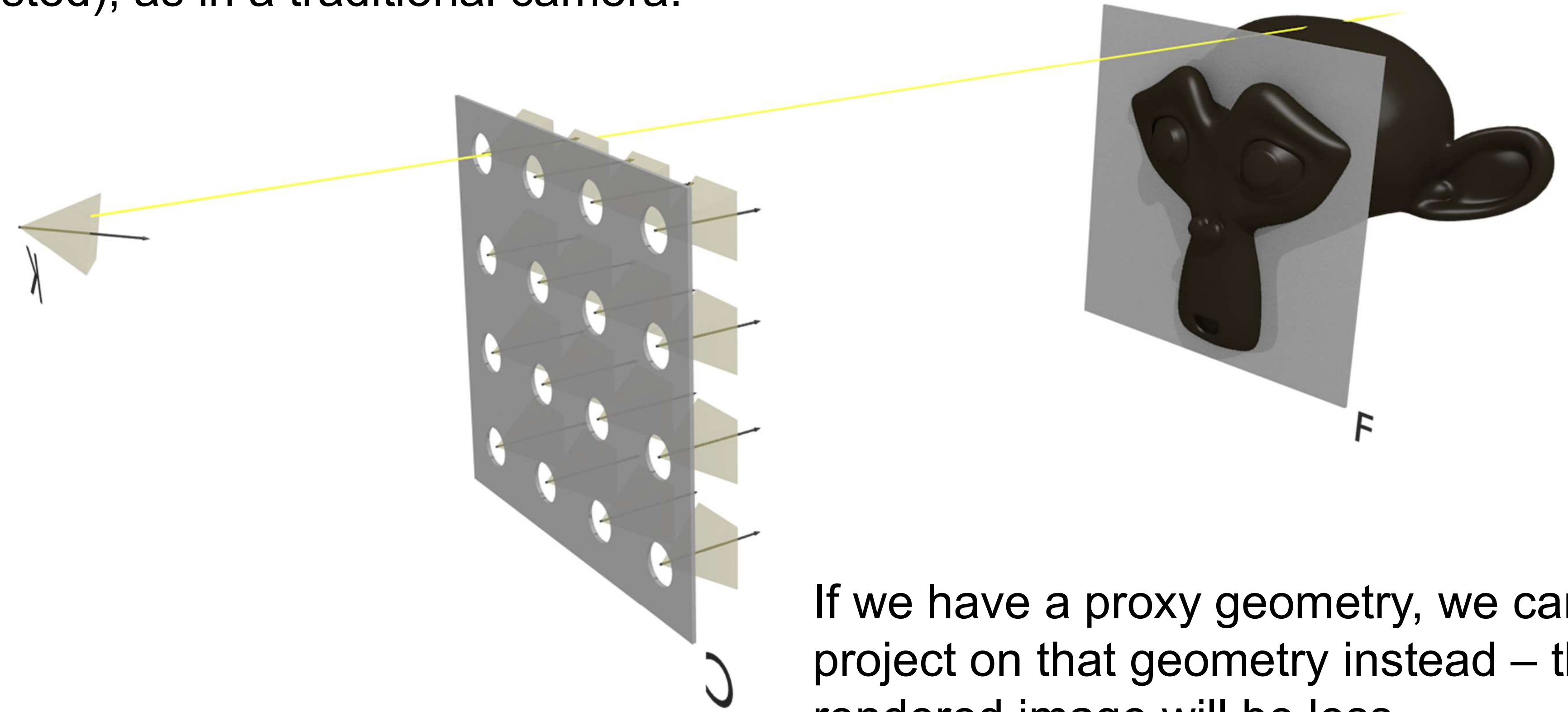


Each camera in the array provides accurate light measurements only for the rays originating from its pinhole aperture.

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

Light field rendering (3/3)

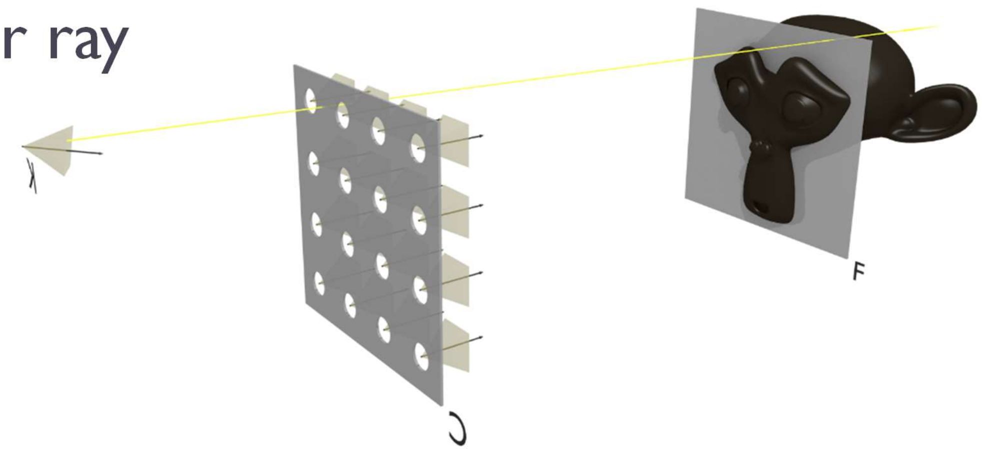
The rays from the camera need to be projected on the focal plane F. The objects on the focal plane will be sharp, and the objects in front or behind that plane will be blurry (ghosted), as in a traditional camera.



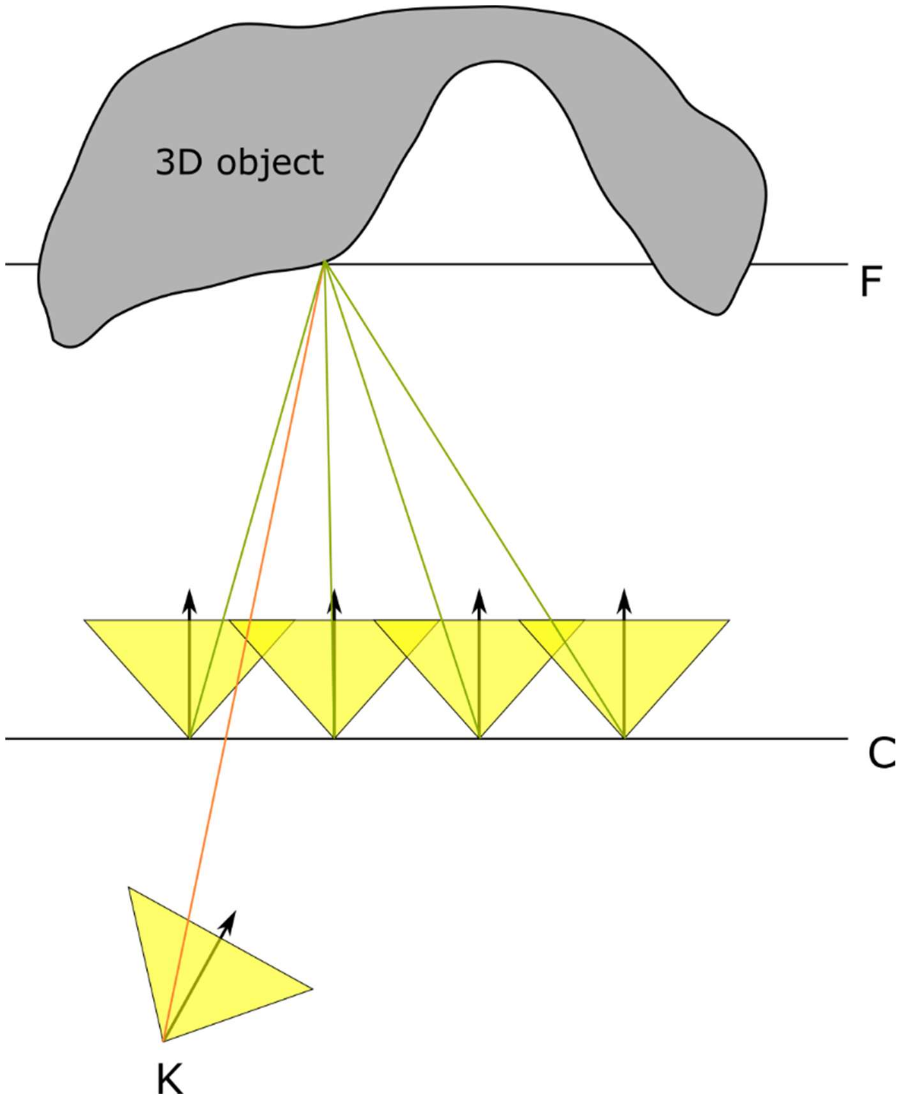
If we have a proxy geometry, we can project on that geometry instead – the rendered image will be less ghosted/blurry

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 virual 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

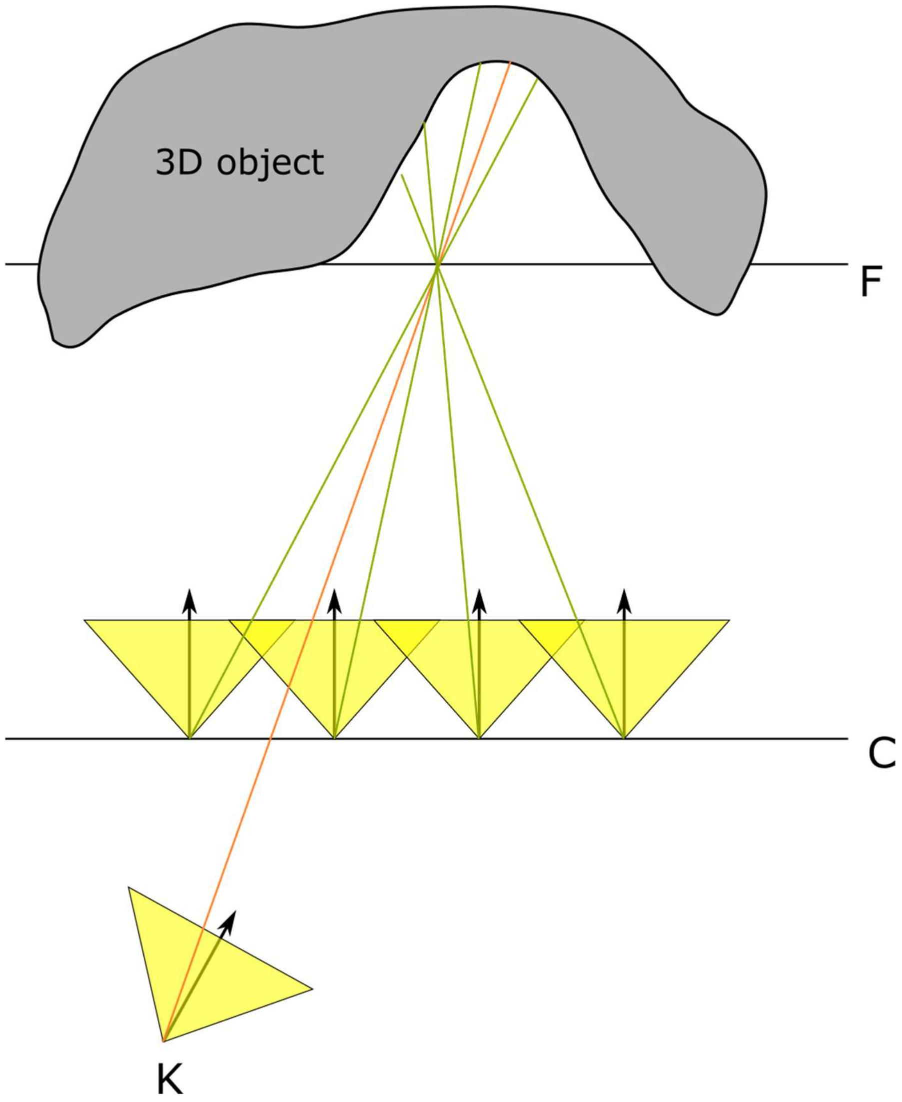


LF rendering – focal plane



- 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 camaras 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

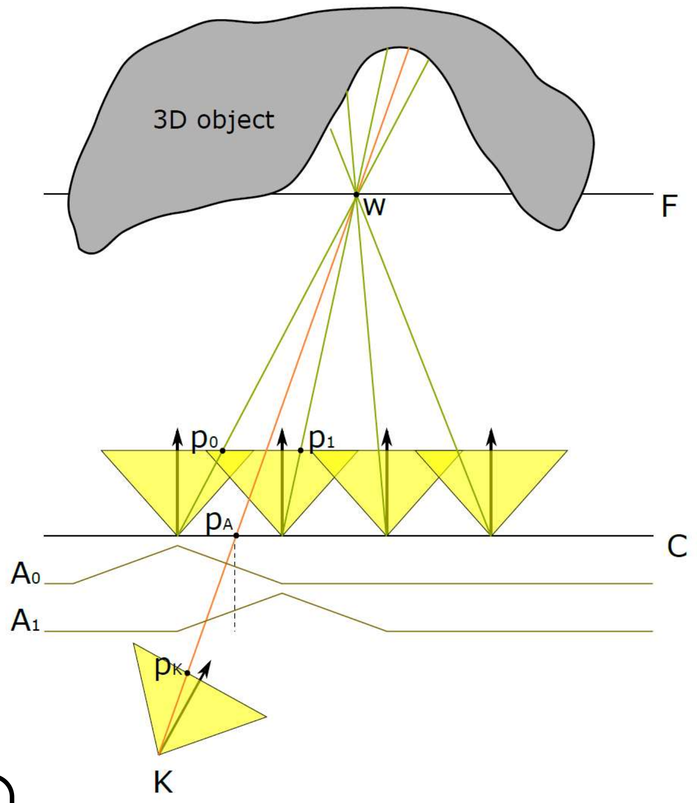
Finding homographic transformation 1/3

- For the pixel coordinates $\boldsymbol{p}_k$ of the virtual camera K, we want to find the corresponding coordinates $\boldsymbol{p}_i$ in the camera array image
- Given the world 3D coordinates of a point $\boldsymbol{w}$:

$$\boldsymbol{p}_i = \boldsymbol{K}\,\boldsymbol{P}\,\boldsymbol{V}_i\,\boldsymbol{w}$$

Intrinsic camera matrix ($\boldsymbol{K}$) — Projection matrix ($\boldsymbol{P}$) — View matrix ($\boldsymbol{V}_i$)

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



Finding homographic transformation 2/3

- A homography between two views is usually found as:

$$\boldsymbol{p}_K = \boldsymbol{K}_K \boldsymbol{P} \boldsymbol{V}_K \boldsymbol{w}$$
$$\boldsymbol{p}_i = \boldsymbol{K}_i \boldsymbol{P} \boldsymbol{V}_i \boldsymbol{w}$$

hence

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

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

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

Finding homographic … n 3/3

The plane in
the camera coordinates
(not world coordinates)

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

$$\underbrace{\begin{bmatrix} x_i \\ y_i \\ d_i \\ w_i \end{bmatrix}}_{\hat{\boldsymbol{p}}_i} = \underbrace{\begin{bmatrix} f_x & 0 & 0 & c_x \\ 0 & f_y & 0 & c_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\hat{\boldsymbol{K}}_i} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ n_x^{(c)} & n_y^{(c)} & n_z^{(c)} & -\boldsymbol{N}^{(\boldsymbol{c})} \cdot \boldsymbol{w}_F^{(c)} \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\hat{\boldsymbol{P}}} \underbrace{\begin{bmatrix} v_{11} & v_{12} & v_{13} & v_{14} \\ v_{21} & v_{22} & v_{23} & v_{24} \\ v_{31} & v_{32} & v_{33} & v_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{V}_i} \underbrace{\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}}_{\boldsymbol{w}}$$

- Where d_i is the distance to the plane (set to 0)
- Hence

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

References

- Light fields
  - Micro-lens array

    - Ng, Ren and Levoy, Marc and Bredif, M. and D., & Gene and Horowitz, Mark and Hanrahan, P. (2005). *Light field photography with a hand-held plenoptic camera*.

  - Camera array

    - OVERBECK, R.S., ERICKSON, D., EVANGELAKOS, D., PHARR, M., AND DEBEVEC, P. 2018. A system for acquiring, processing, and rendering panoramic light field stills for virtual reality. *ACM Transactions on Graphics 37*, 6, 1–15.
    - ISAKSEN, A., MCMILLAN, L., AND GORTLER, S.J. 2000. Dynamically reparameterized light fields. *Proc of SIGGRAPH '00*, ACM Press, 297–306.