Image Formation and Camera Model

Kuk-Jin Yoon

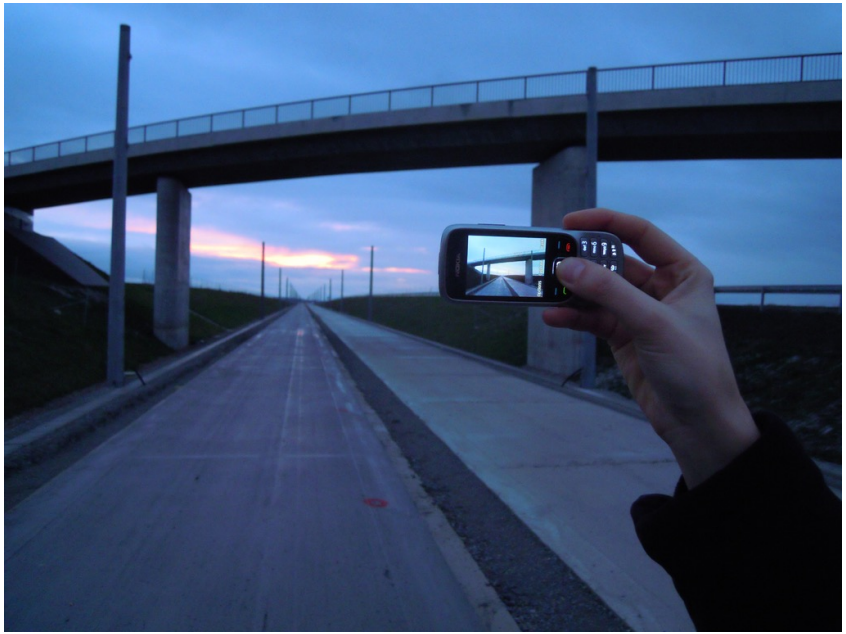
Visual Intelligence Lab.
Department of Mechanical Engineering

KAIST

What is an Image?

- Image: **projection** of the 3D world onto an (2D) image plane
  - 2-dimensional patterns of brightness values
  - Formed by the **projection** of 3D objects

"Dürer - Man Drawing a Lute" in Geometrie (1535) by Albrecht Dürer



"Image created with a mobile phone" by Olaf Simons - Own work.

KAIST

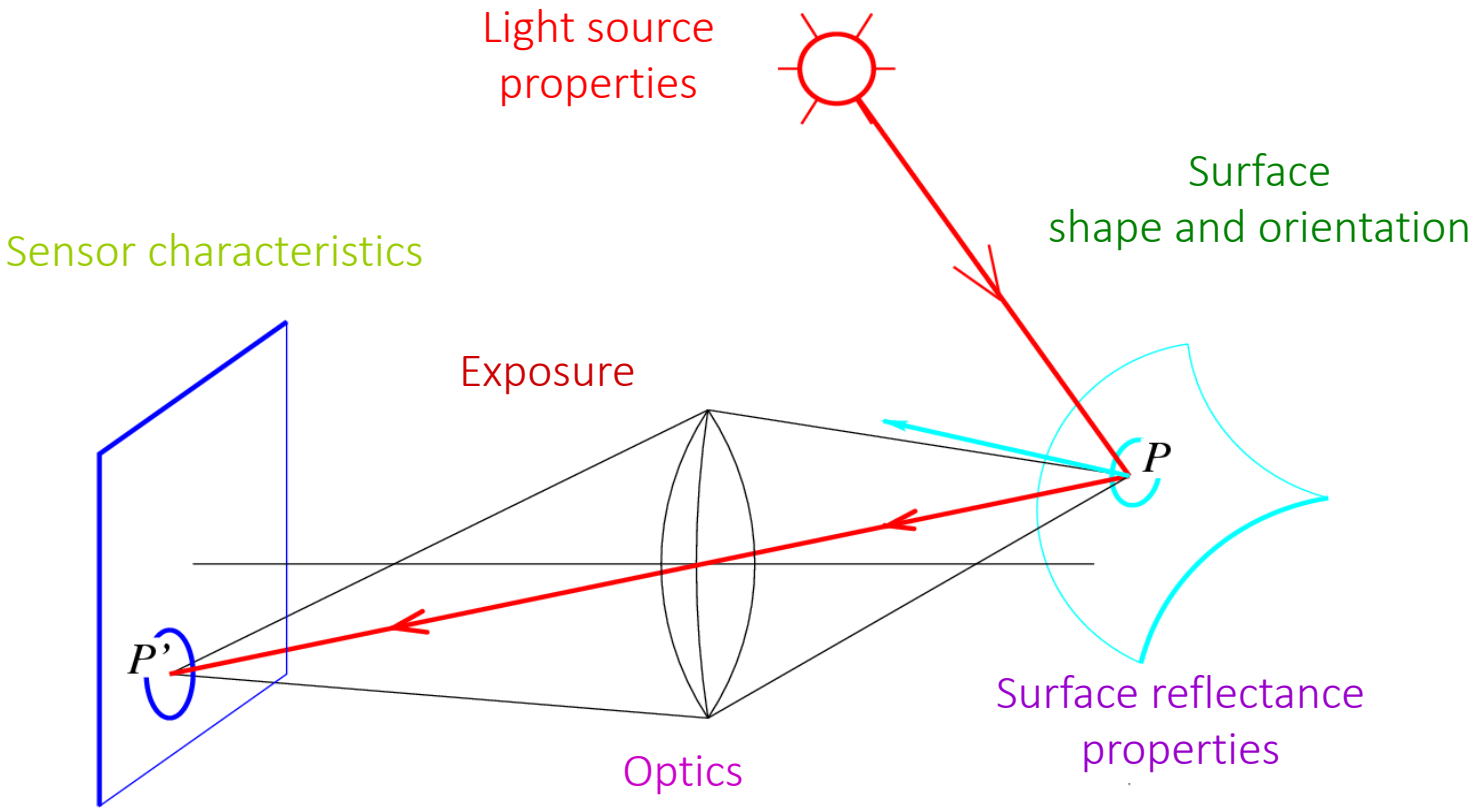
This is how we obtain images

https://static.makeuseof.com/wp-content/uploads/2018/05/iphone-camera-settings-670x335.jpg

https://www.aolor.com/images/tutorial/devices/take-photos-with-iphone.jpg

KAIST

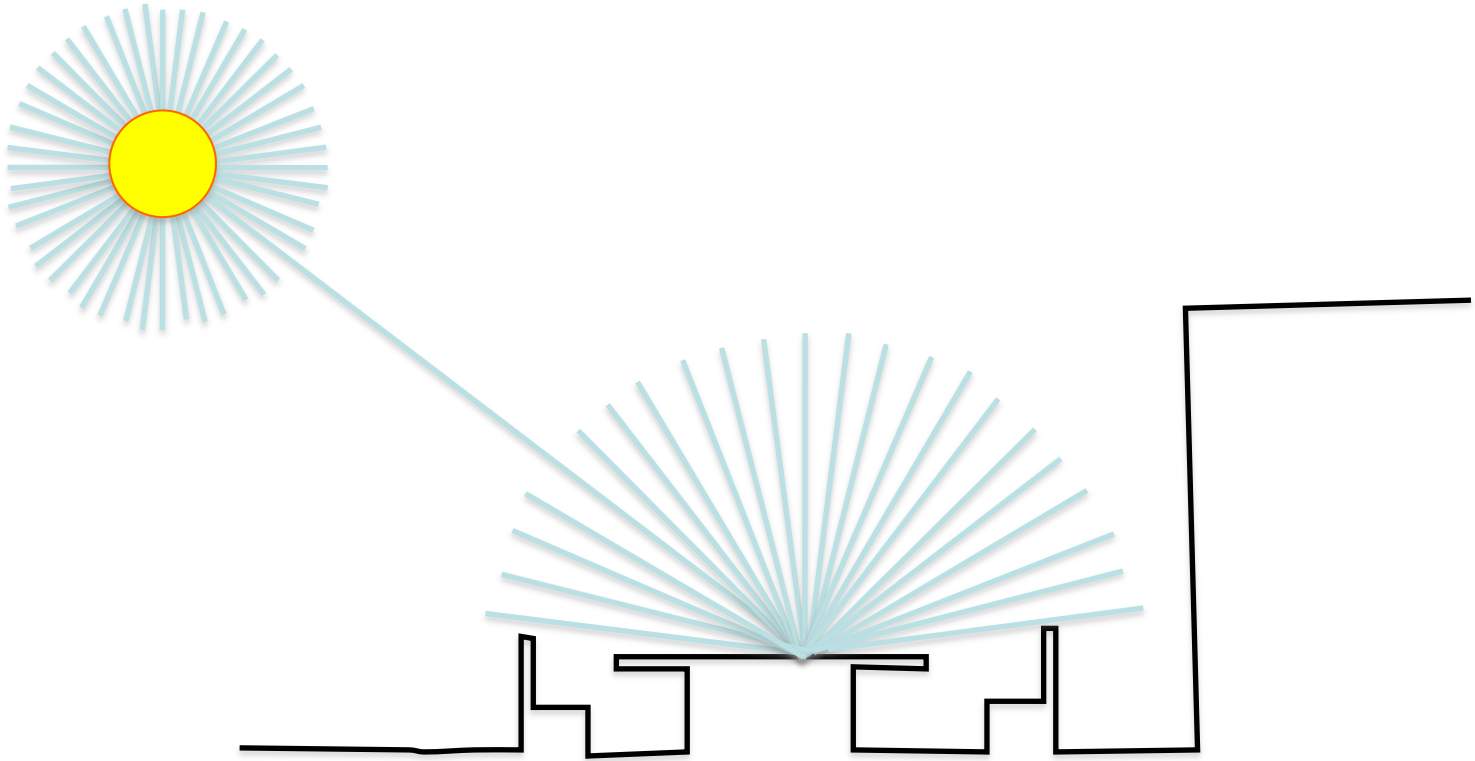
Image Formation



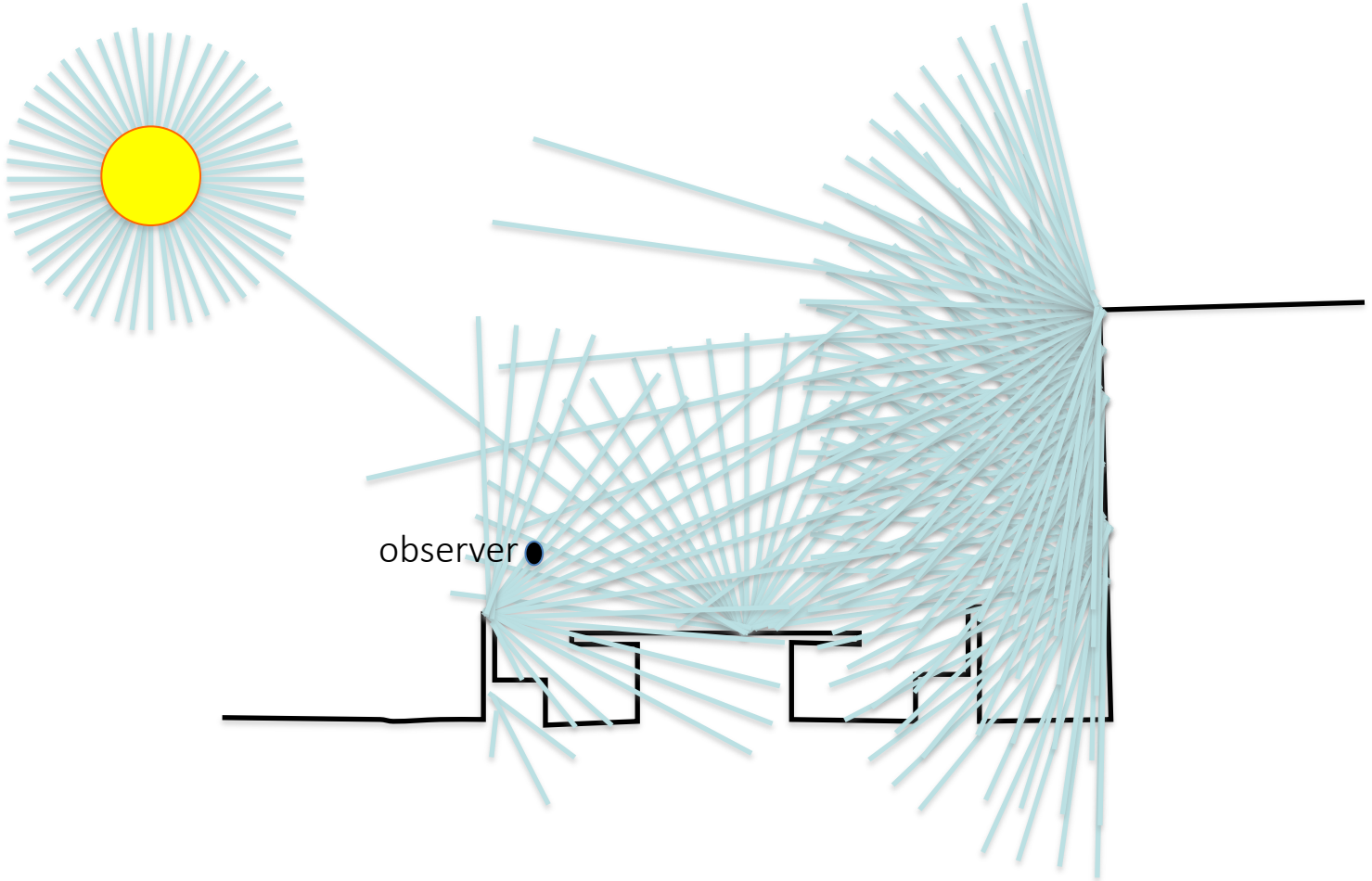
Cameras

- The pinhole projection model
  - Qualitative properties
  - Perspective projection matrix
- Cameras with lenses
- Digital cameras

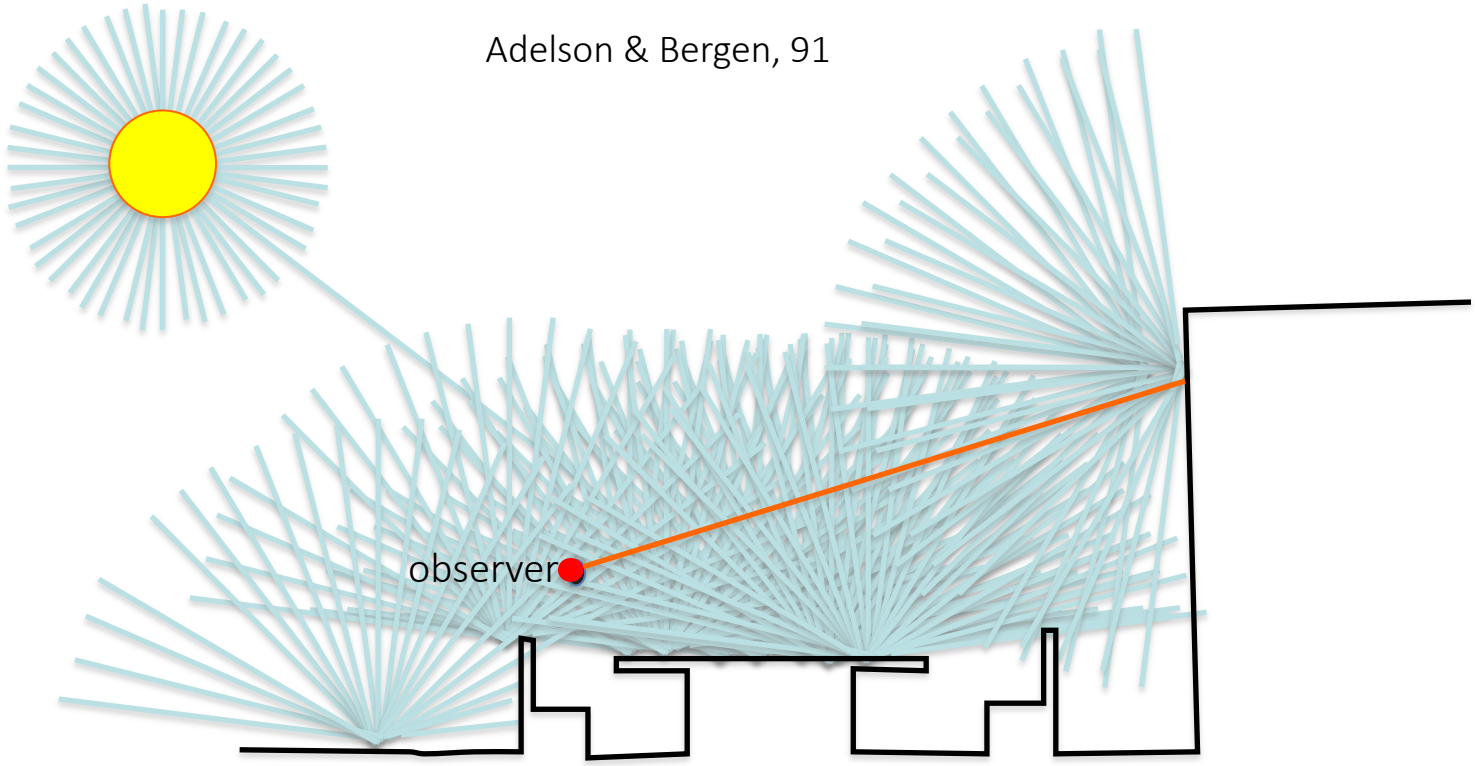
Structure of Ambient Light



Structure of Ambient Light



Plenoptic Function



The intensity P can be parameterized as:

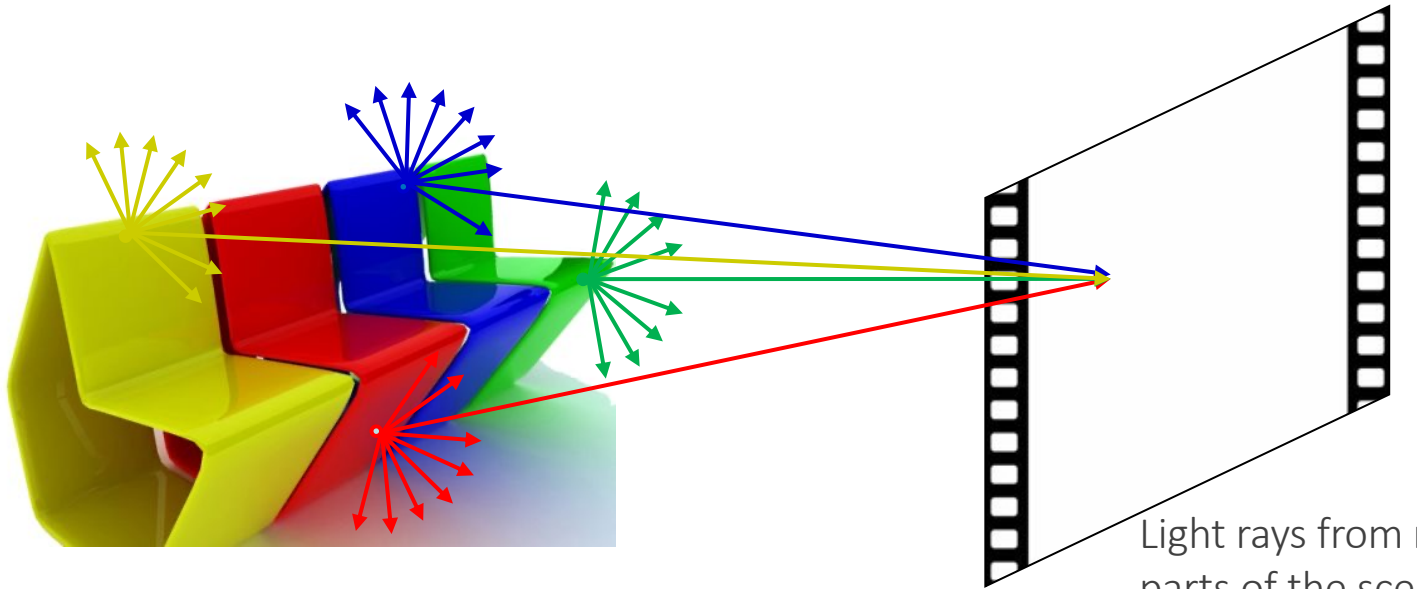
P (q, f, l, t, X, Y, Z)

- 3D for viewpoint (X, Y, Z)
- 2D for ray direction (q, f)
- 1D for wavelength (colors) (l)
- 1D for time (t)

KAIST

Designing a Camera

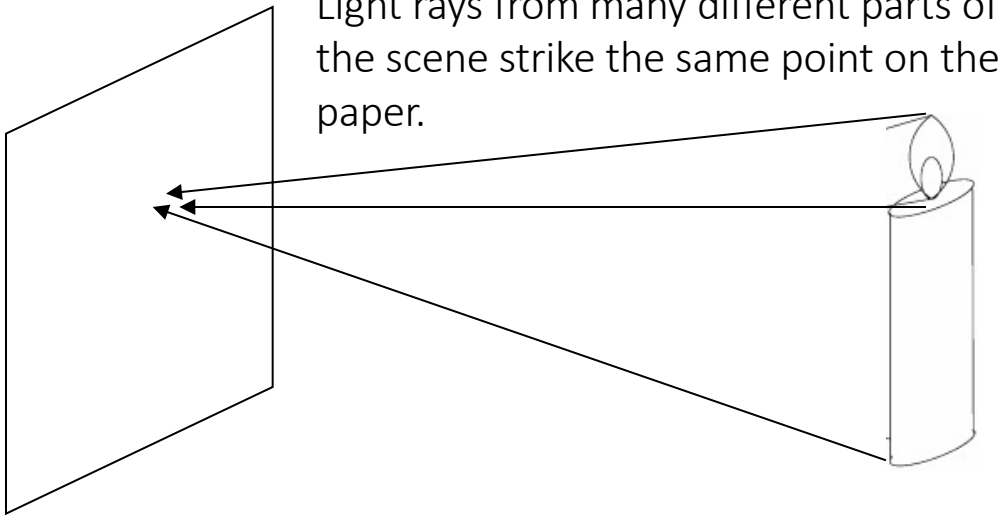
- Putting a piece of film in front of an object



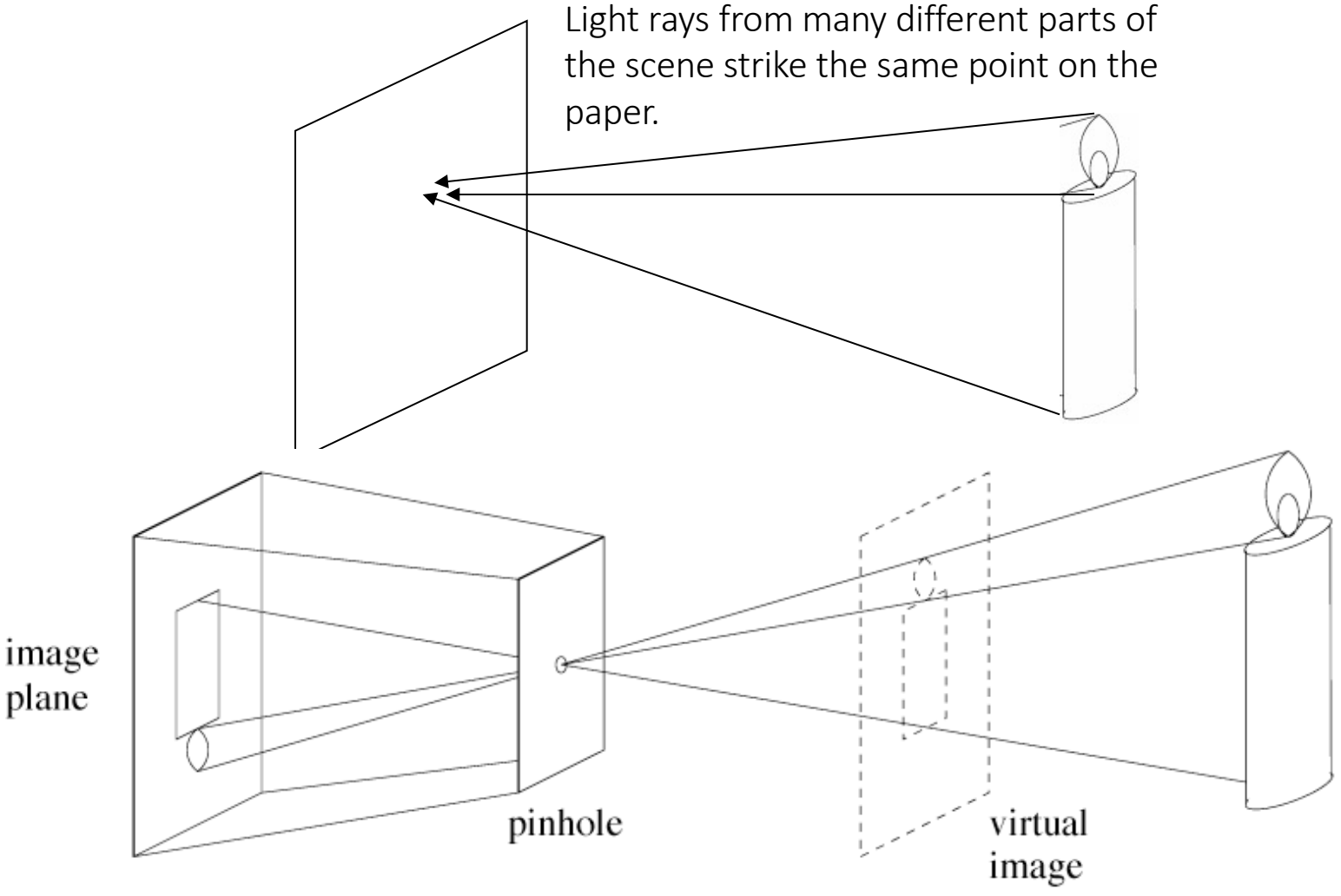
Why is there no picture appearing on the film?

Light rays from many different parts of the scene strike the same point on the film.

Light rays from many different parts of the scene strike the same point on the paper.



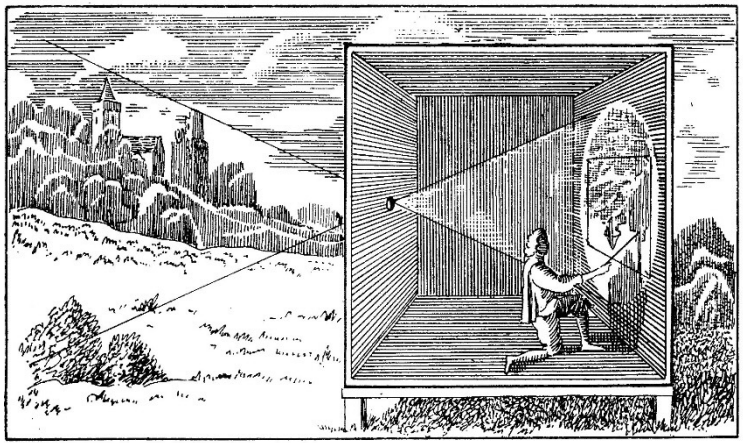
Light rays from many different parts of the scene strike the same point on the paper.

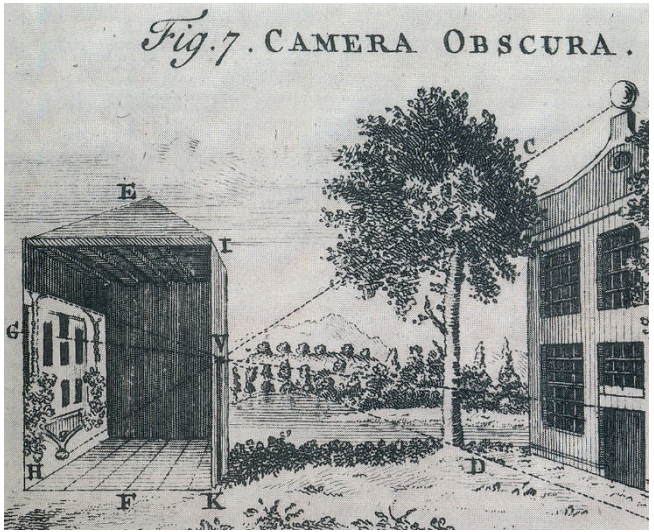


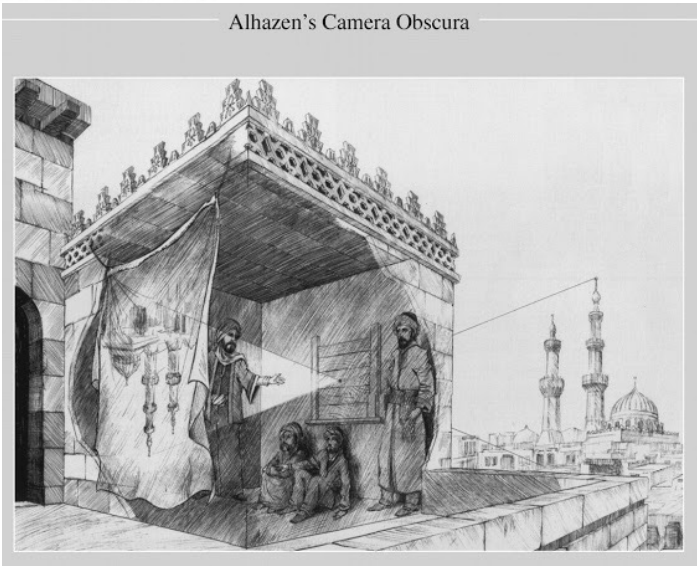
The pinhole camera only allows rays from one point in the scene to strike each point of the paper.

Forsyth & Ponce

Camera Obscura





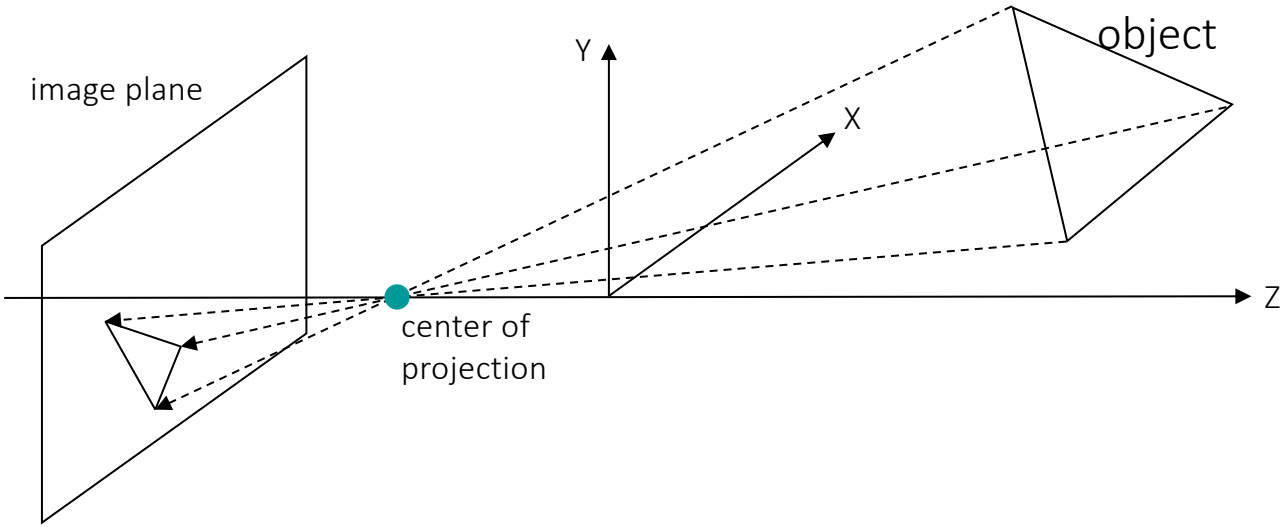


Constructed in 1946 and remodeled to resemble a giant camera in 1957

KAIST

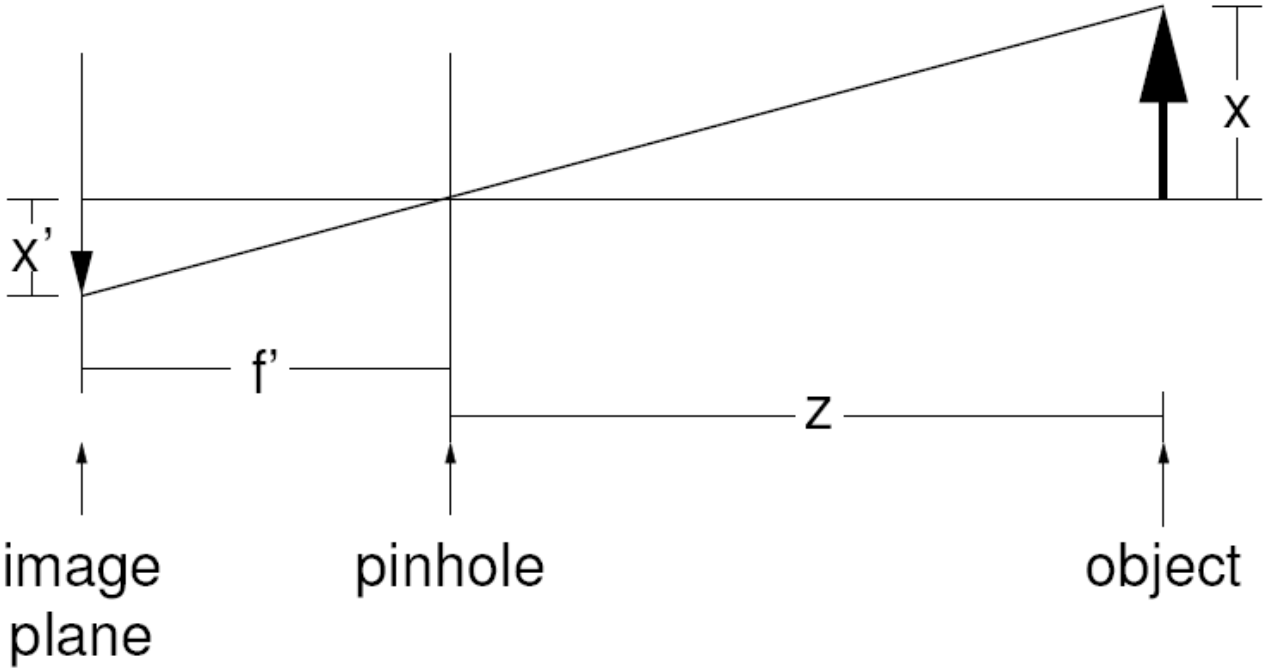
Pinhole Camera Model

- Pinhole model
  - Capturing **pencil of rays**
  - The point is called **center of projection (focal point)**
  - The image is formed on the **image plane**



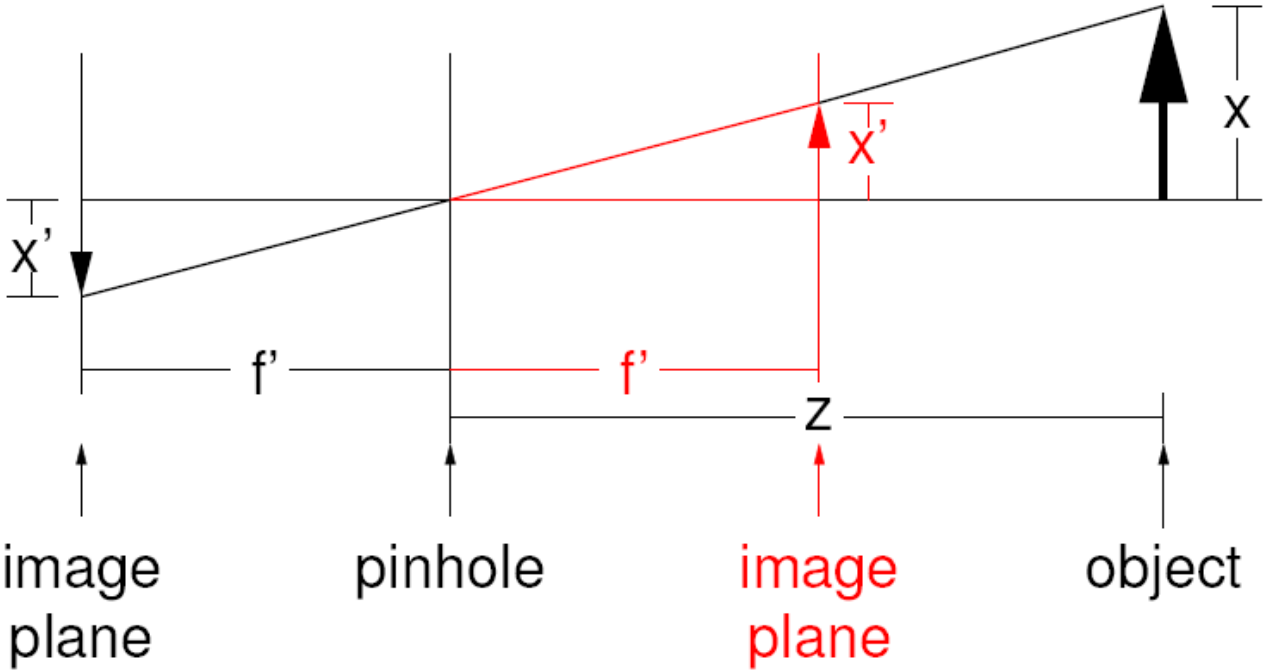
KAIST

Pinhole Camera in 2D



$$X' = (f' / Z)\, X$$

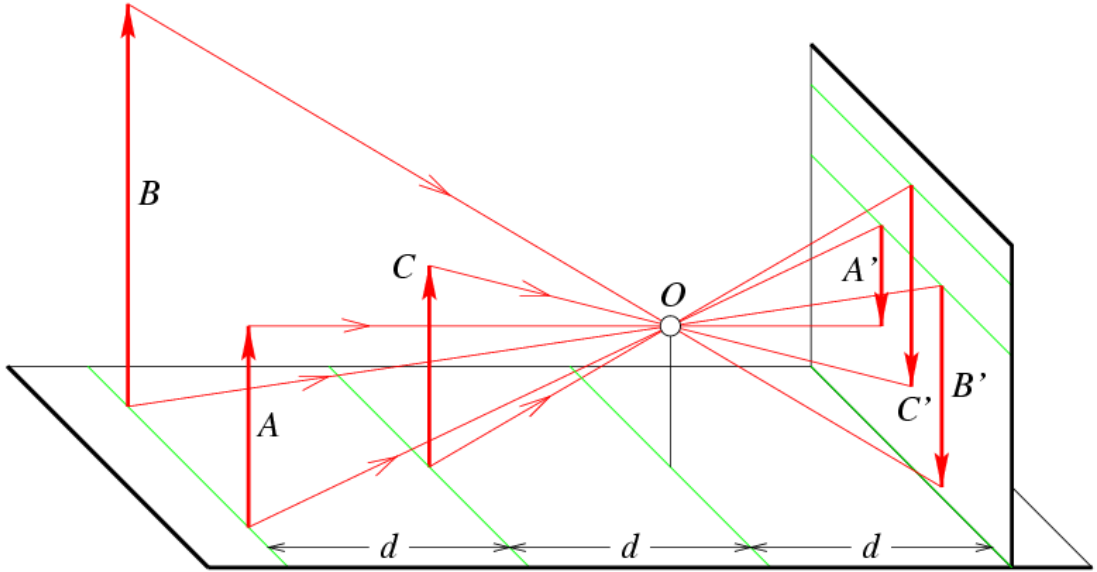
Pinhole Camera in 2D (with Reflected Image Plane)



The image is the same after reflection of the image plane, except that image is the right way up!

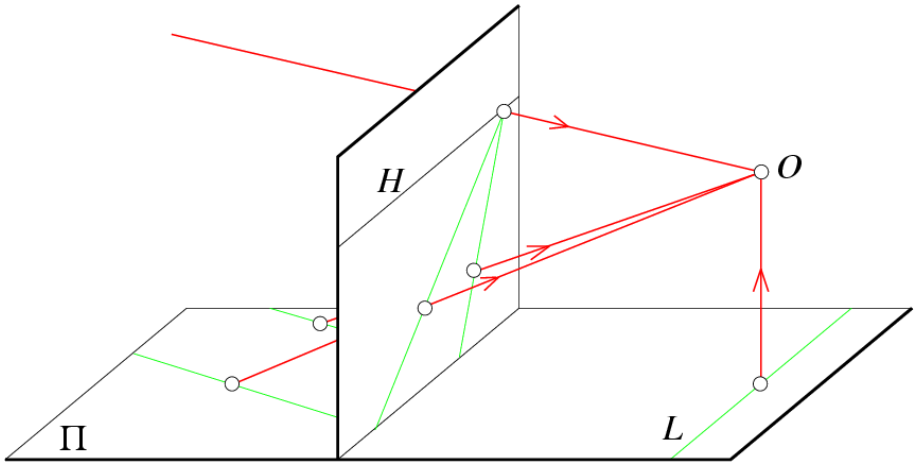
Properties of Projection

- Distant objects appear smaller.
  - The size is inversely proportional to distance.



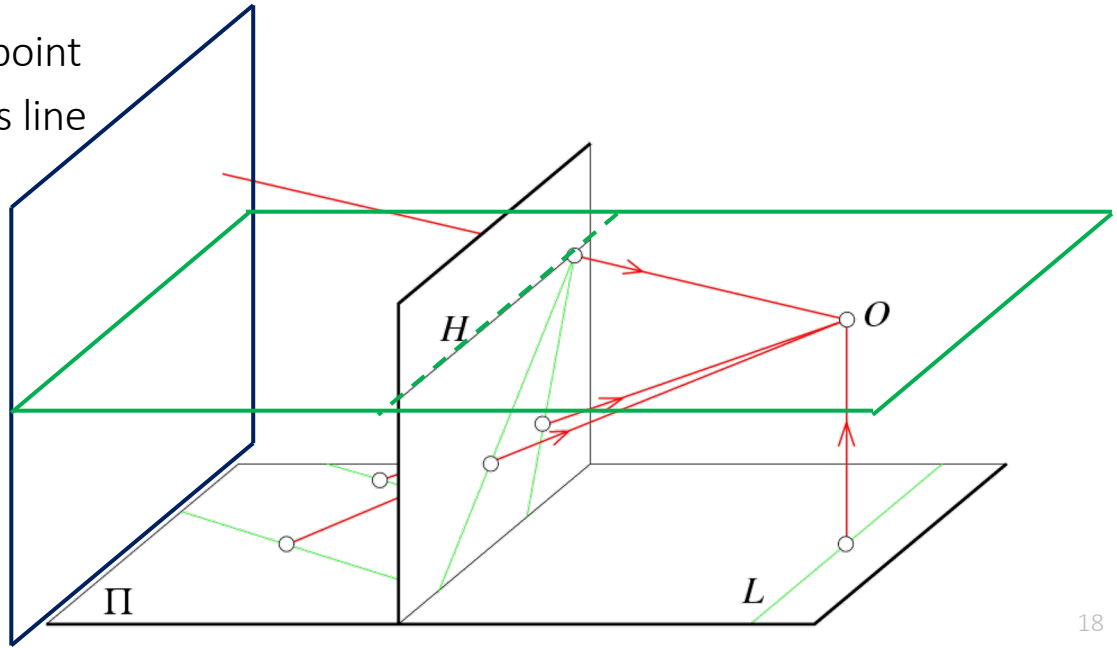
Properties of Projection

- Parallel lines converge at a vanishing point.
  - Each direction in space has its own vanishing point
    - To different directions correspond different vanishing points.
    - But parallels parallel to the image plane remain parallel.



Geometric Properties of Projection

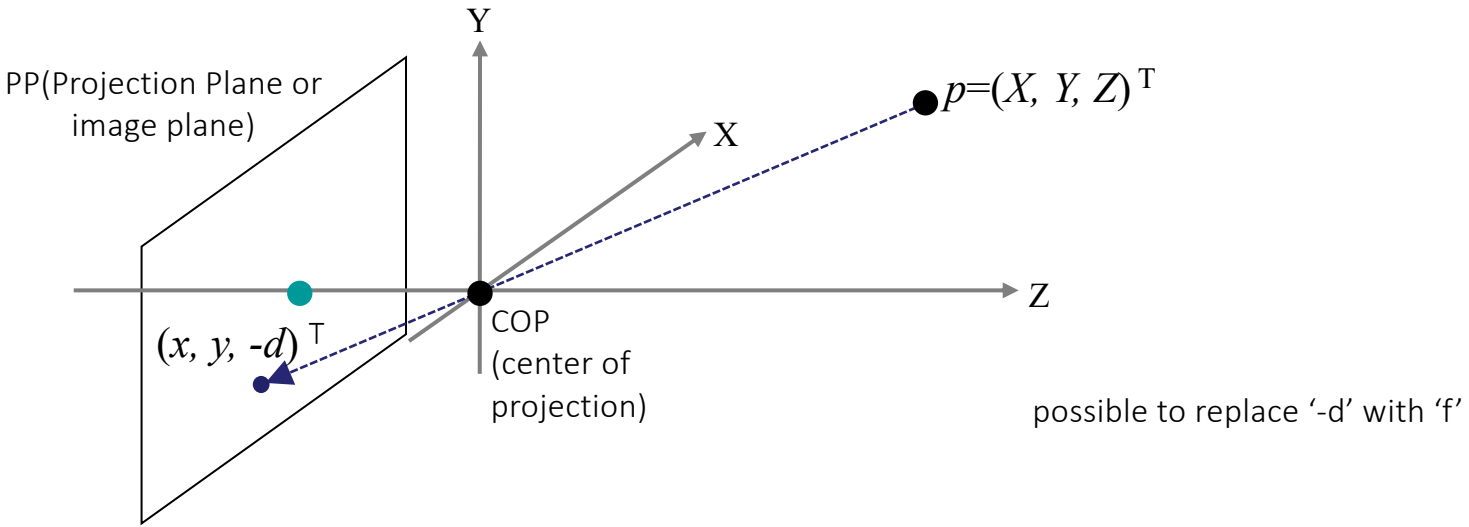
- Points go to points
- Lines go to lines
- Planes go to whole image or half-plane
- Polygons go to polygons
- Degenerate cases:
  - line through focal point yields point
  - plane through focal point yields line



KAIST

Modeling Projection

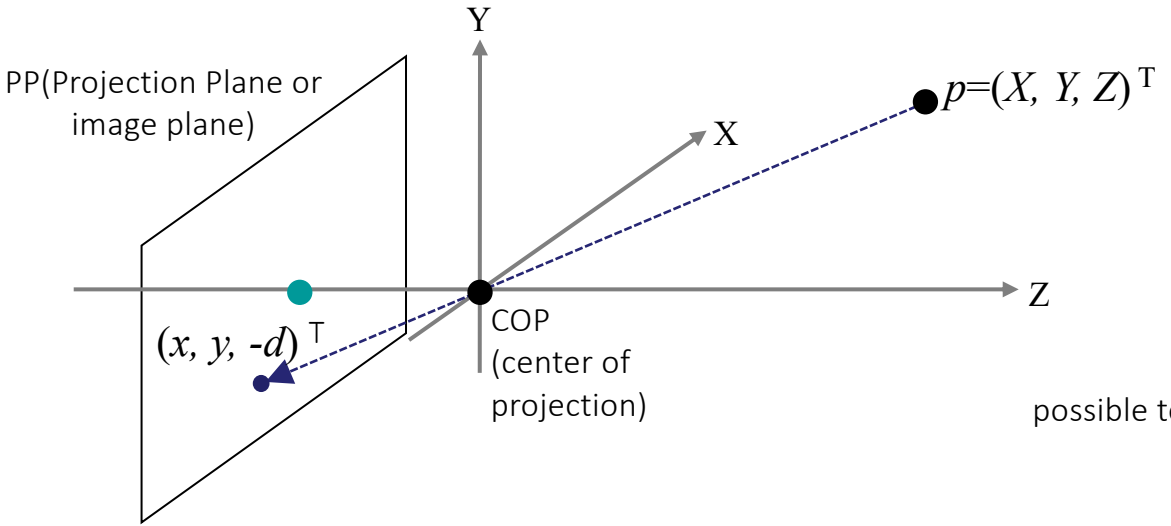
- The coordinate system
  - We will use the pin-hole model as an approximation
  - Put the optical center (**C**enter **O**f **P**rojection) at the origin
  - Put the image plane (**P**rojection **P**lane) *behind* the COP



Modeling Projection

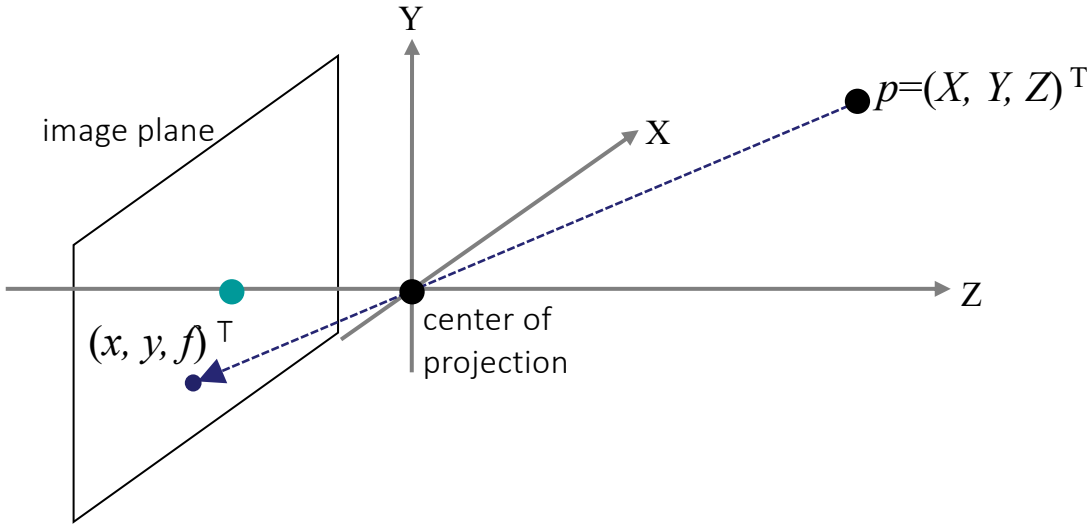
- Projection equations
  - Compute intersection with PP of ray from (X, Y, Z) to COP
  - Derived using similar triangles (on board) $(X,Y,Z) \longrightarrow \left(-d\frac{X}{Z}, -d\frac{Y}{Z}, -d\right)$
    - We get the projection by throwing out the last coordinate:

$$(X,Y,Z) \longrightarrow \left(-d\frac{X}{Z}, -d\frac{Y}{Z}\right) \text{ or } \left(f\frac{X}{Z}, f\frac{Y}{Z}\right)$$



possible to replace '-d' with 'f'

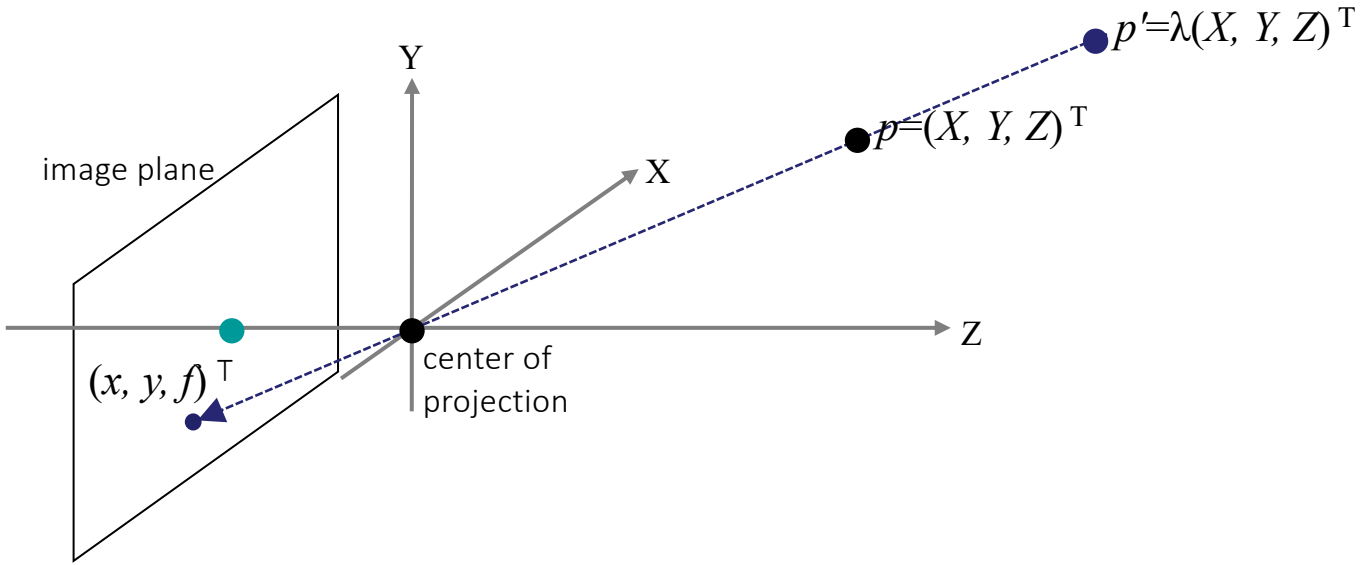
Perspective Projection Modeling



- Projection relation (with the image plane at $z=f$)

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f\dfrac{X}{Z} \\ f\dfrac{Y}{Z} \end{bmatrix} = f \begin{bmatrix} \dfrac{X}{Z} \\ \dfrac{Y}{Z} \end{bmatrix}$$

Perspective Projection Modeling



- Projection equations (with the image plane at $z=f$)

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f\dfrac{\lambda X}{\lambda Z} \\ f\dfrac{\lambda Y}{\lambda Z} \end{bmatrix} = f \begin{bmatrix} \dfrac{X}{Z} \\ \dfrac{Y}{Z} \end{bmatrix}$$

Homogeneous Coordinates

- In the projection process, two different points (X, Y, Z) and (λ X, λ Y, λ Z) (λ ≠0) are mapped onto the same point in the image plane.
  - Given a point (x, y) on the Euclidean plane, and choosing a non-zero real number Z, the triple (xZ, yZ, Z) is called a set of homogeneous coordinates for the point.
  - Projective (or homogeneous) coordinates of a point in projective space

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad (x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

  - Converting from homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w) \qquad \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Rightarrow (x/w, y/w, z/w)$$

Perspective Projection

- Projection is a matrix multiply using homogeneous coordinates:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1/d & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ -z/d \end{bmatrix} \Rightarrow (-d\frac{x}{z}, \quad -d\frac{y}{z})$$

divide by the third coordinate

This is known as perspective projection

- The matrix is the projection matrix

Perspective Projection

- How does scaling the projection matrix change the transformation?

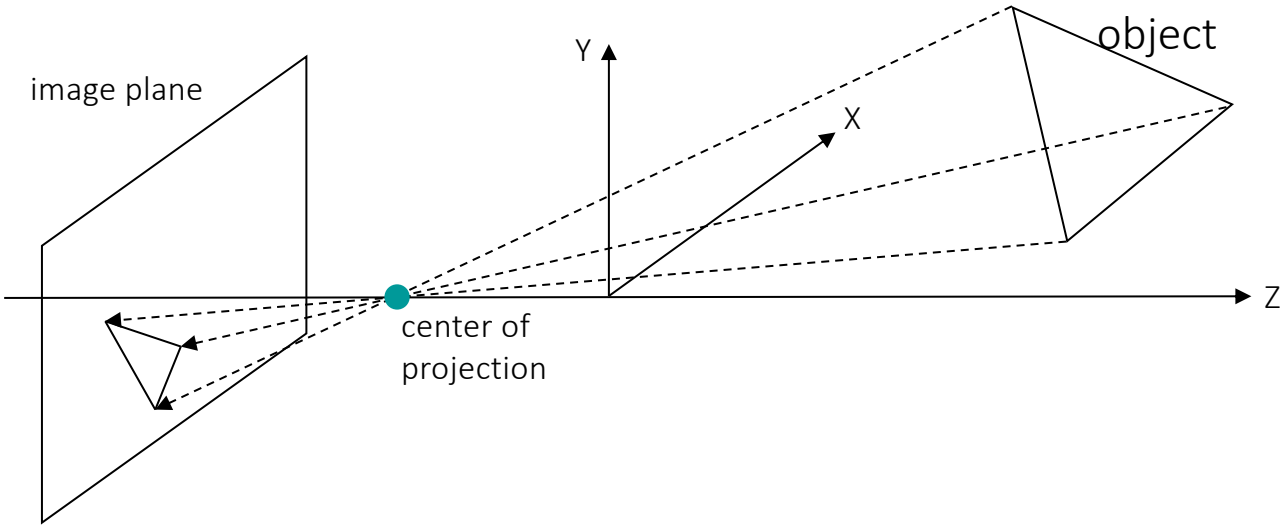
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1/d & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ -z/d \end{bmatrix} \Rightarrow (-d\frac{x}{z}, \quad -d\frac{y}{z})$$

$$\begin{bmatrix} -d & 0 & 0 & 0 \\ 0 & -d & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} -dx \\ -dy \\ z \end{bmatrix} \Rightarrow (-d\frac{x}{z}, \quad -d\frac{y}{z})$$

CAMERA MODEL

Pinhole Camera Model

- Pinhole model
  - Capturing **pencil of rays**
  - The point is called **center of projection (focal point)**
  - The image is formed on the **image plane**

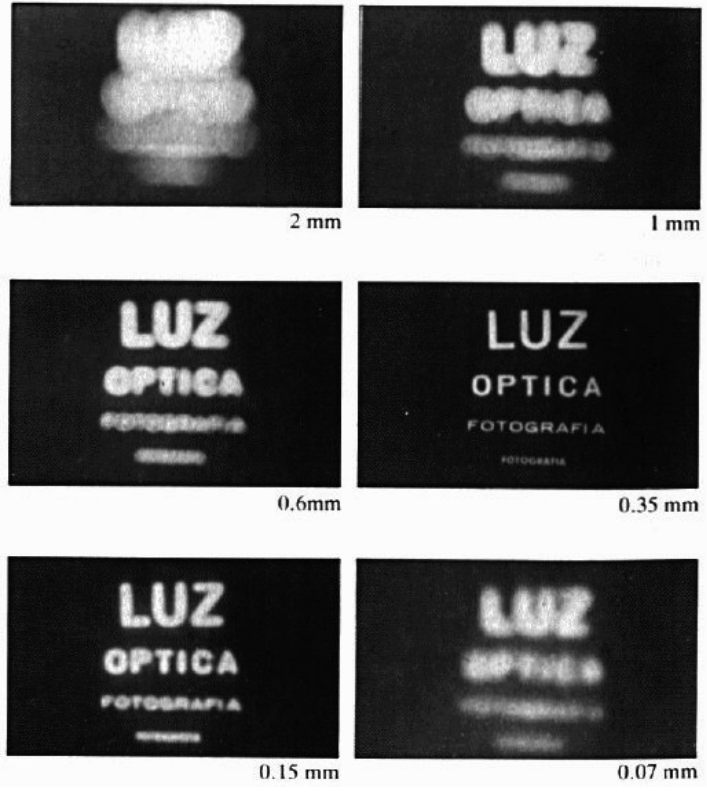


KAIST

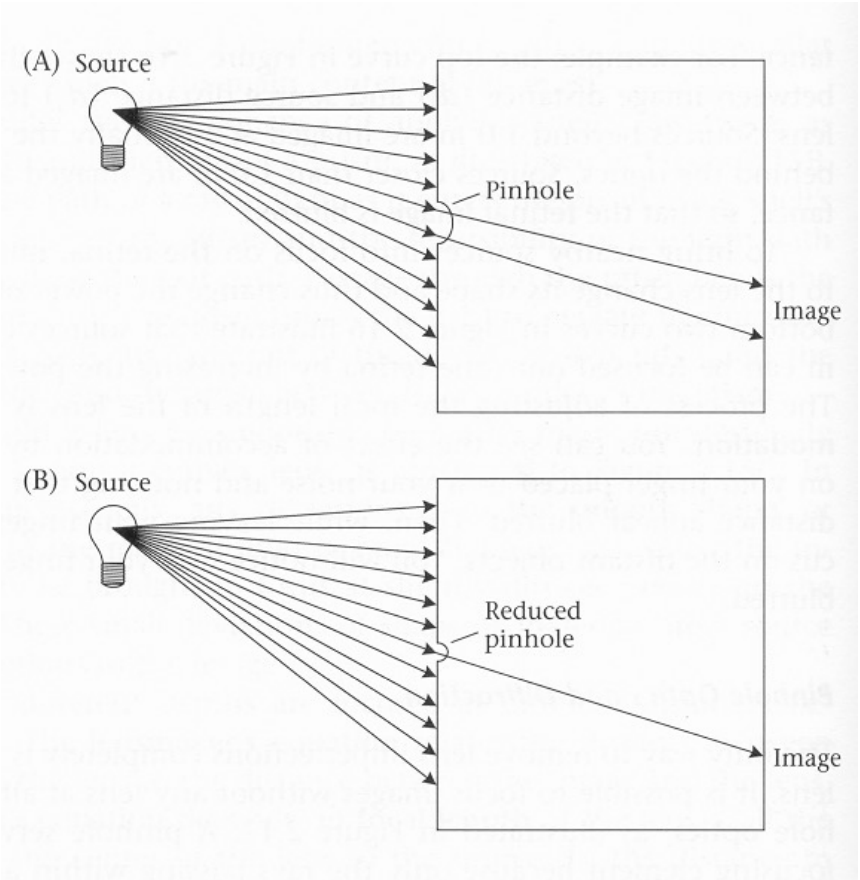
Why Not Use Pinhole Cameras?

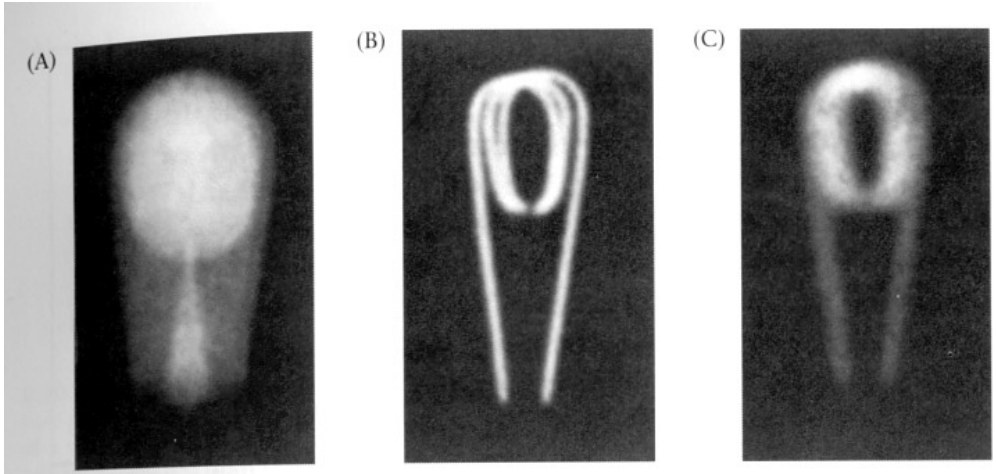
- Pinhole size
  - too big: many directions are averaged, blurring the image.
  - too small: diffraction effects blur the image.

- Generally, pinhole cameras are dark, because a very small set of rays from a particular point hits the screen.



Effect of Pinhole Size

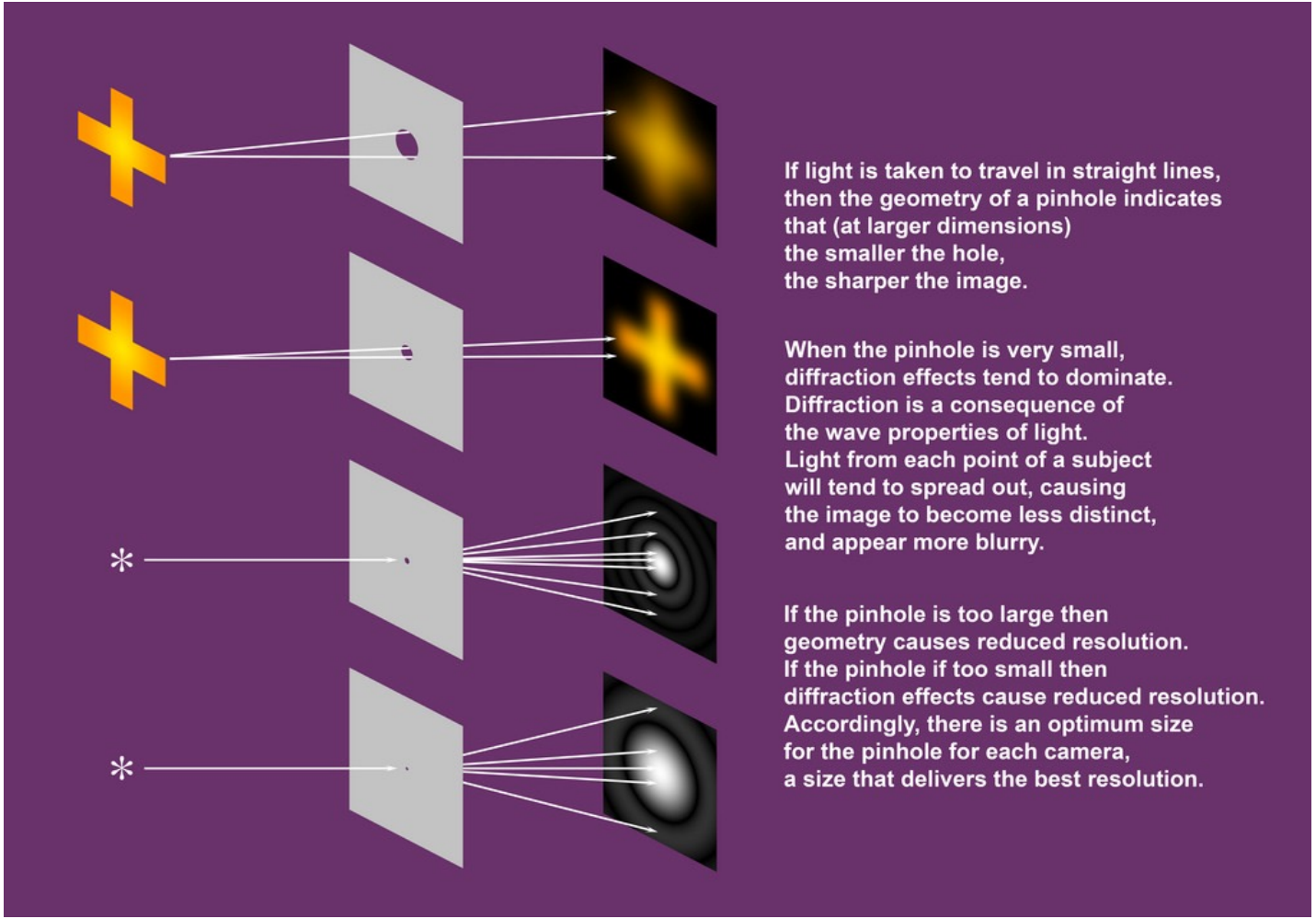




2.18 DIFFRACTION LIMITS THE QUALITY OF PINHOLE OPTICS. These three images of a bulb filament were made using pinholes with decreasing size. (A) When the pinhole is relatively large, the image rays are not properly converged, and the image is blurred. (B) Reducing the size of the pinhole improves the focus. (C) Reducing the size of the pinhole further worsens the focus, due to diffraction. From Ruechardt, 1958.

Wandell, Foundations of Vision, Sinauer, 1995

Effect of Pinhole Size

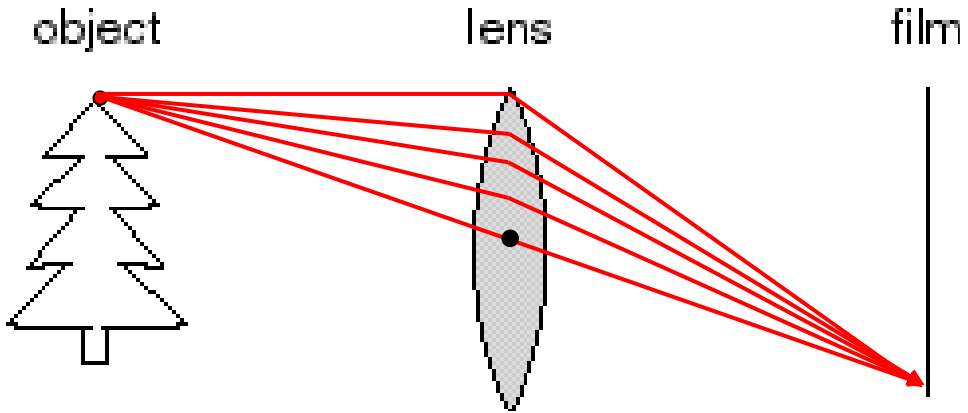


Pinhole Images

Exposure 4 seconds

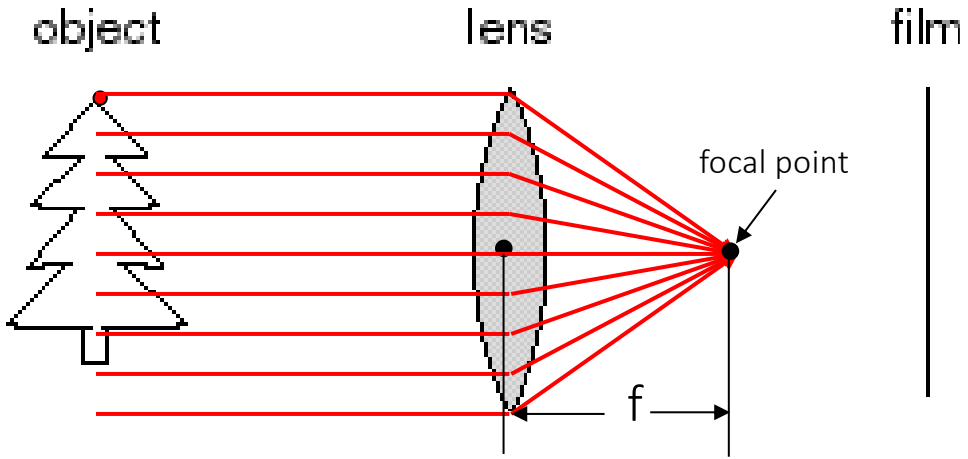
Exposure 96 minutes

Adding a Lens



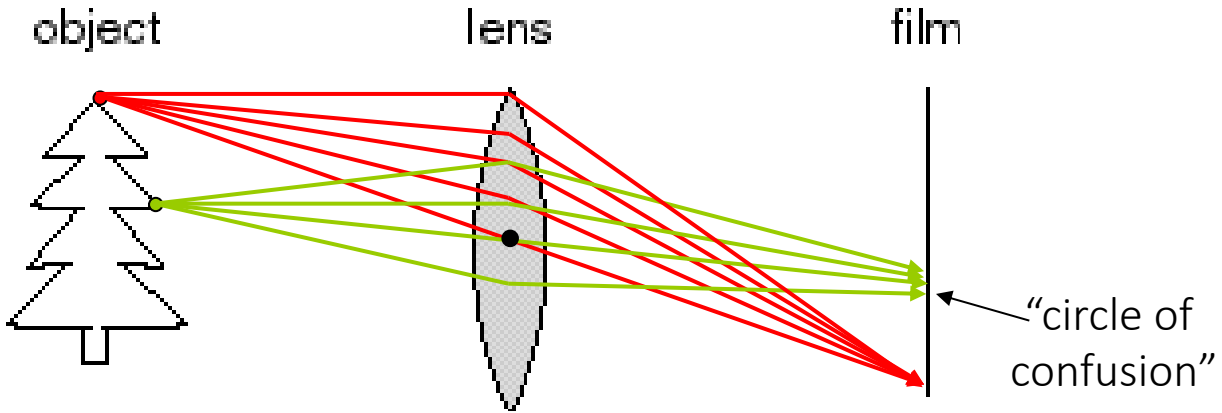
- A lens focuses light onto the film
  - Rays passing through the center are not deviated

Adding a Lens



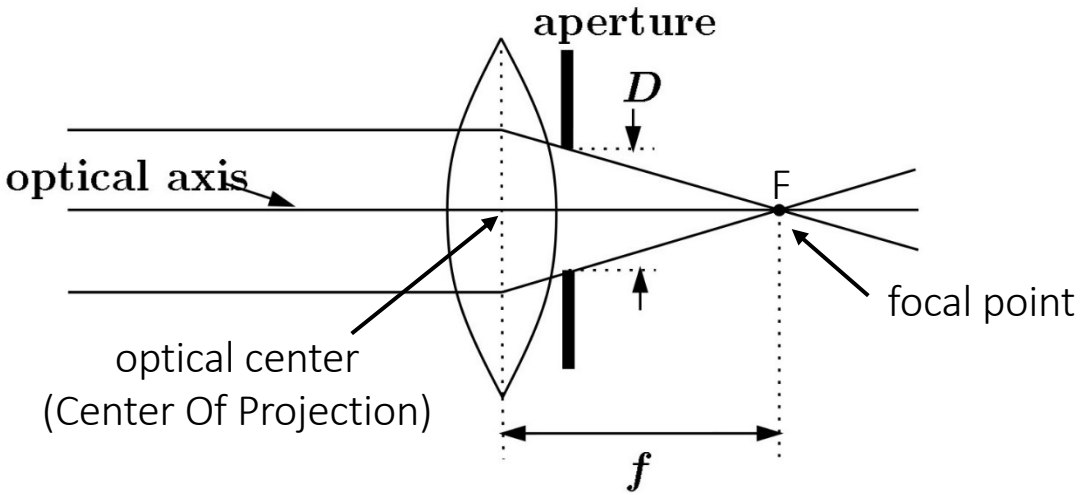
- A lens focuses light onto the film
  - Rays passing through the center are not deviated
  - All parallel rays converge to one point on a plane located at the *focal length f*

Adding a Lens



- A lens focuses light onto the film
  - There is a specific distance at which objects are “in focus”
    - other points project to a “circle of confusion” in the image

Lenses

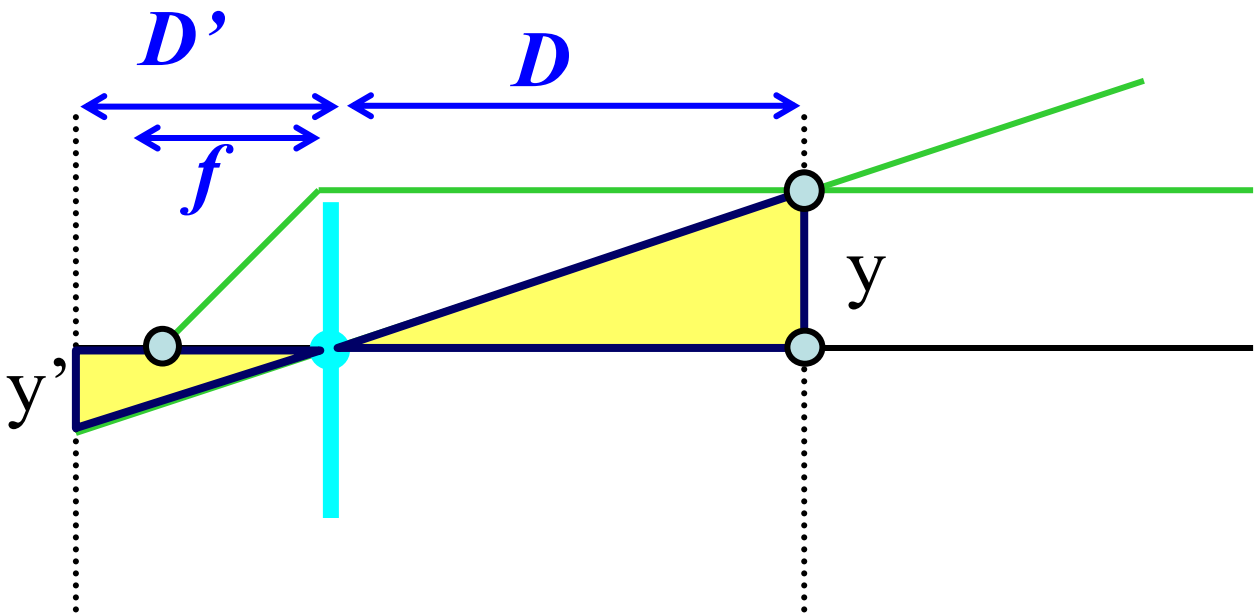


- A lens focuses parallel rays onto a single focal point
  - focal point at a distance *f* beyond the plane of the lens
    - *f* is a function of the shape and index of refraction of the lens
  - Aperture of diameter D restricts the range of rays
    - aperture may be on either side of the lens
  - Lenses are typically spherical (easier to produce)

Thin Lens Formula

Similar triangles everywhere!

$$y'/y = D'/D$$

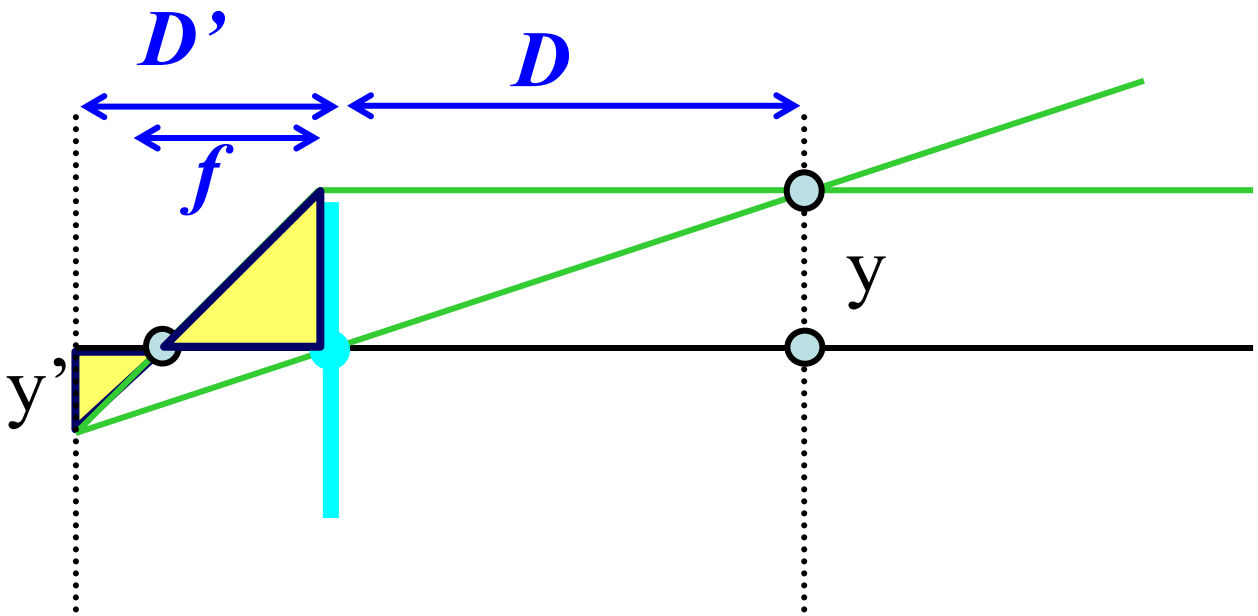


Thin Lens Formula

Similar triangles everywhere!

$$y'/y = D'/D$$

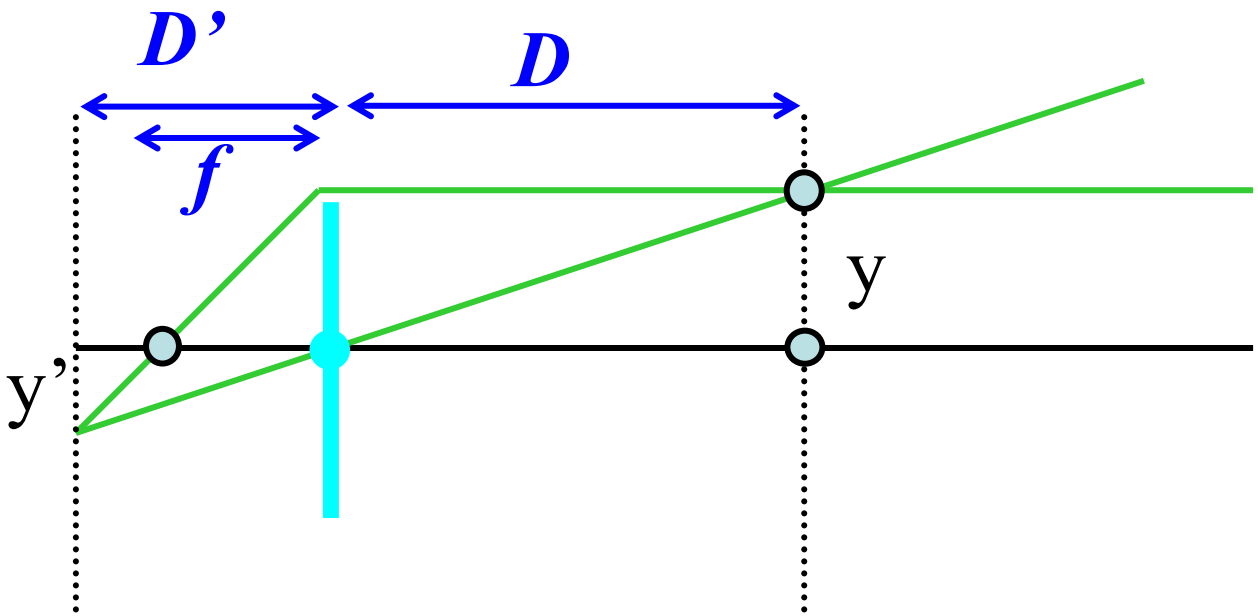
$$y'/y = (D'-f)/f$$



Thin Lens Formula

$$\frac{1}{D'} + \frac{1}{D} = \frac{1}{f}$$

Any point satisfying the thin lens equation is in focus.



Pinhole Model with a Single Lens

- A lens follows the pinhole model for objects that are in focus.

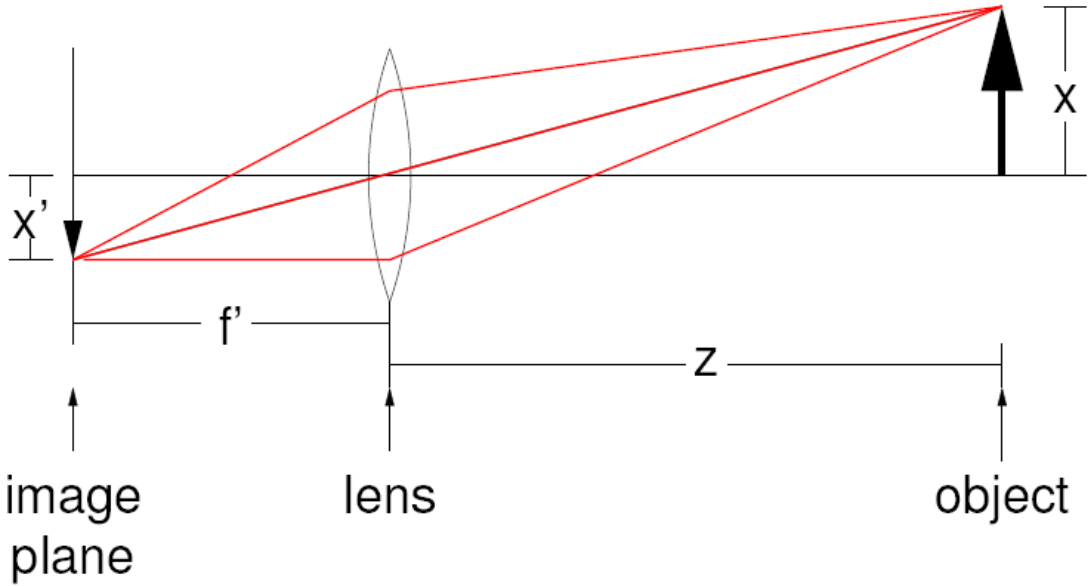
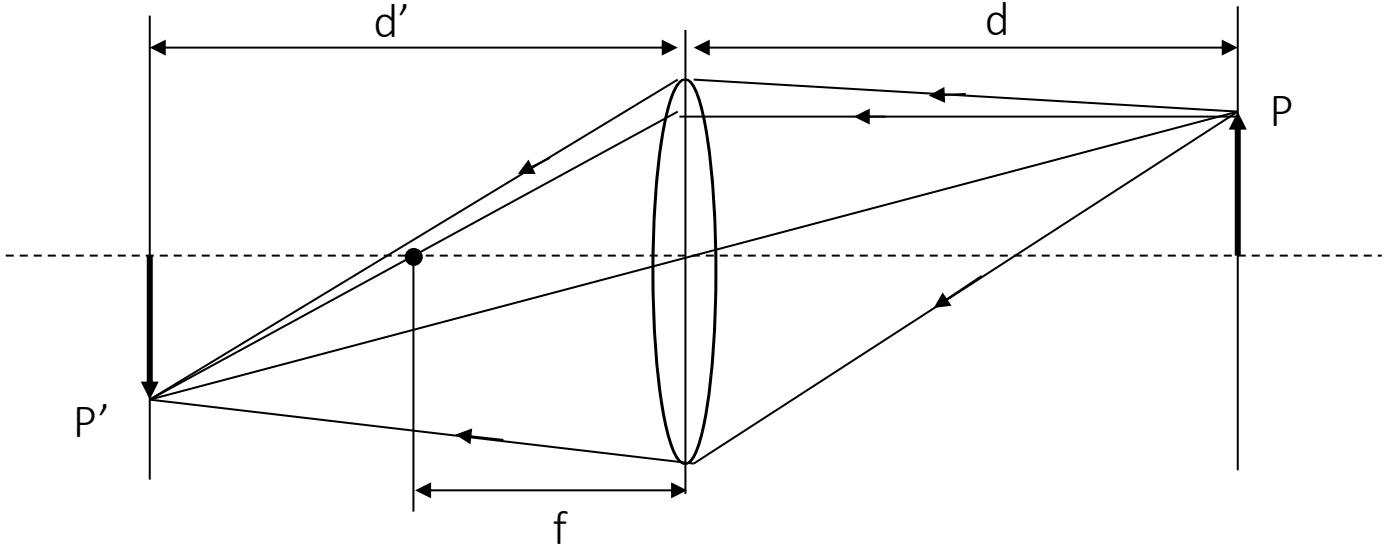


Image Formation using Lenses

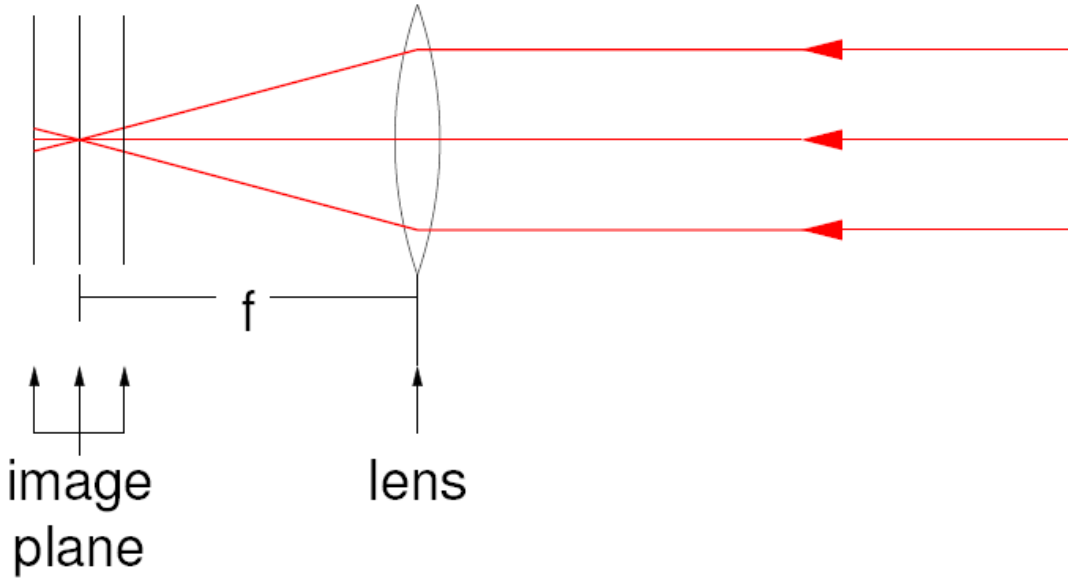
- Lenses are used to avoid problems with pinholes.
- Ideal lens: Same projection as pinhole but gathers more light!



- Gaussian Thin Lens Formula: $\frac{1}{d} + \frac{1}{d'} = \frac{1}{f}$
- f is the focal length of the lens – determines the lens's ability to refract light
- f different from the effective focal length f' discussed before!

Out-of-focus

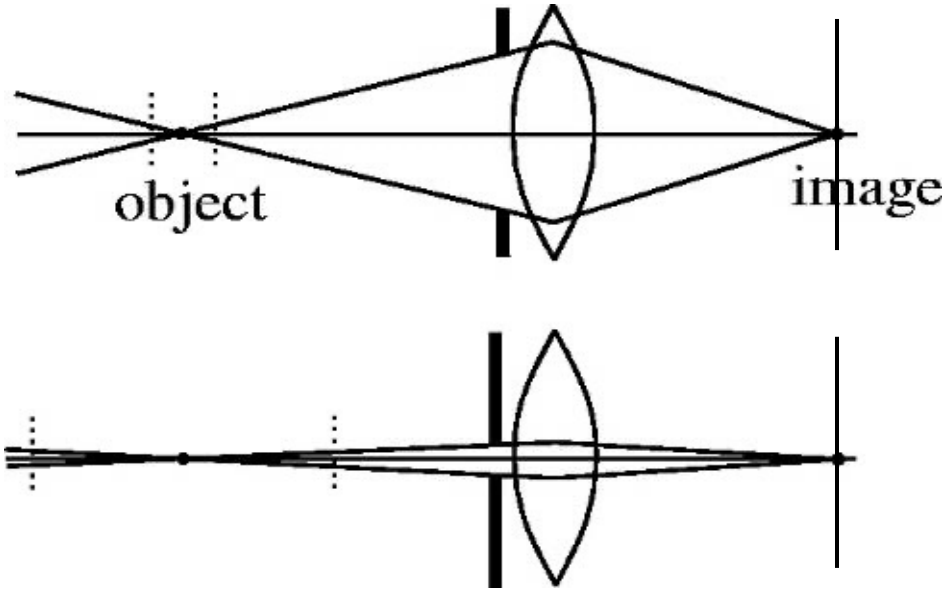
- An image plane at the wrong distance
  - Rays from different parts of the lens create a blurred region (the “point spread function”).



Depth of Field

Controlling the Depth of Field

- Changing the aperture size
  - A smaller aperture increases the range in which the object is approximately in focus.
    - But small aperture reduces amount of light – need to increase exposure



KAIST

F-Number

- The f-number *f*/#, often notated as *N*, is given by

$$f/\# = N = \frac{f}{D}$$

  - *f*: focal length
  - *D*: diameter of the aperture

Effect of Lens Aperture




f/22	f/8
f/4	f/2.8

images from Wikipedia http://en.wikipedia.org/wiki/Depth_of_field

Effect of Lens Aperture

At f/32, the background is distracting.

At f/5.6, the flowers are isolated from the background.

images from Wikipedia http://en.wikipedia.org/wiki/Depth_of_field

KAIST

Field of View

- FOV vs. focal length

Five images using 24, 28, 35, 50 and 72mm equivalent zoom lengths, portrait format, to illustrate angles of view.

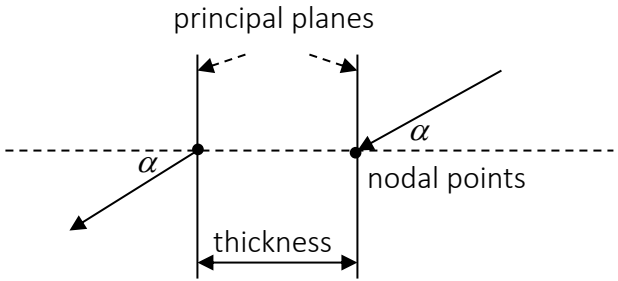
images from Wikipedia http://en.wikipedia.org/wiki/Field_of_view_(image_processing)

KAIST

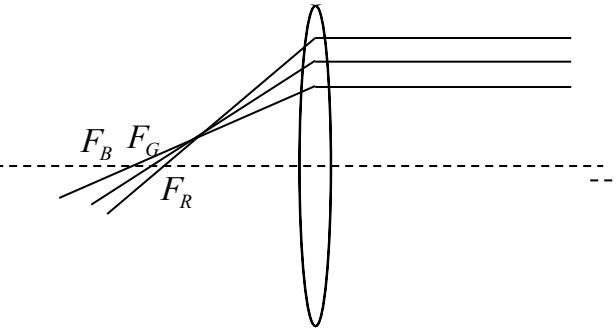
Real Lenses

Compound (Thick) Lens

- Lens flaws
  - Chromatic aberration
  - Spherical aberration
  - Radial distortion
  - …

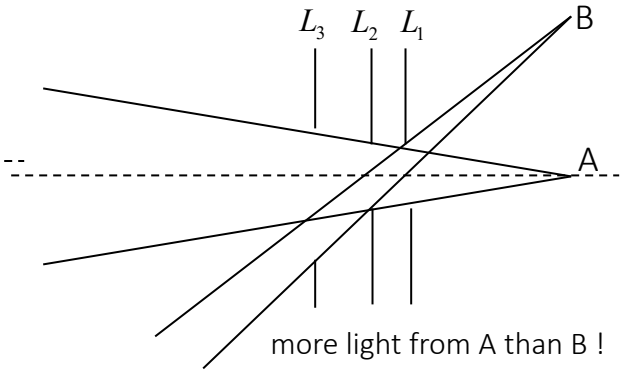


Chromatic Aberration



Lens has different refractive indices for different wavelengths.

Vignetting



more light from A than B !

Radial and Tangential Distortion

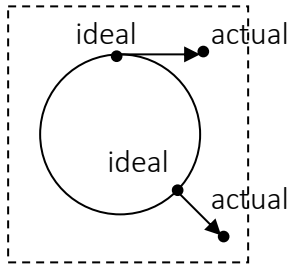


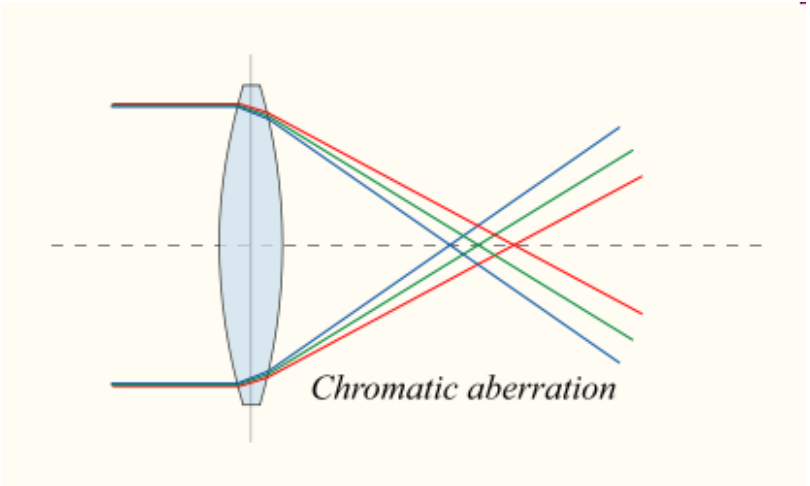
image plane

Lens Glare

- Stray inter-reflections of light within the optical lens system.
- Happens when very bright sources are present in the scene.

Chromatic Aberration

- Lens has different refractive indices for different wavelengths
  - Causing color fringing

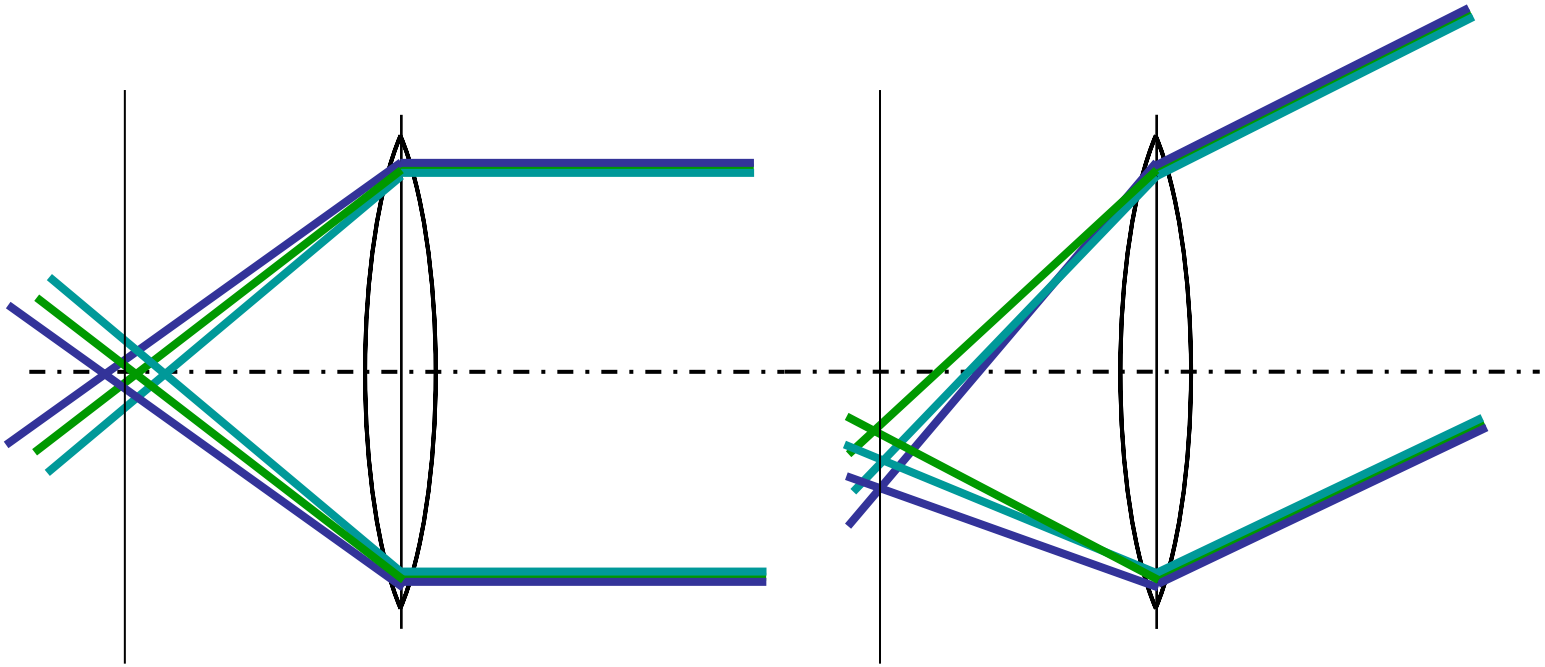


Top: a photo taken with a built-in lens of digital camera (Sony V3).
Bottom: photo taken with the same camera, but with additional wide angle lens. The effect of aberration is visible around the dark edges.

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

KAIST

Chromatic Aberration

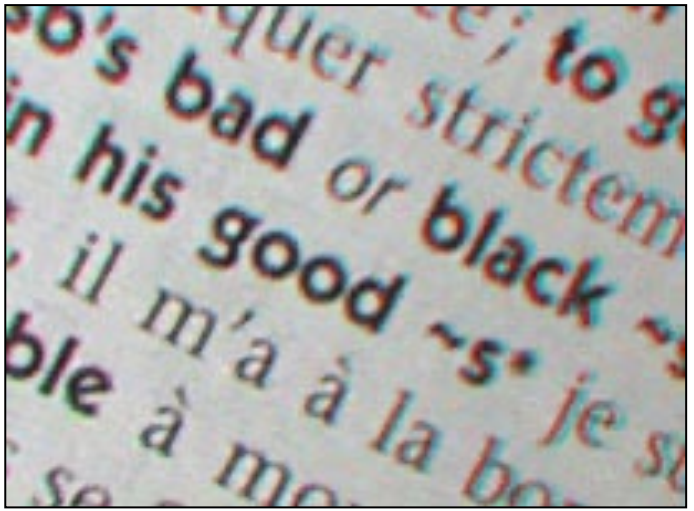


longitudinal chromatic aberration (axial)

transverse chromatic aberration (lateral)

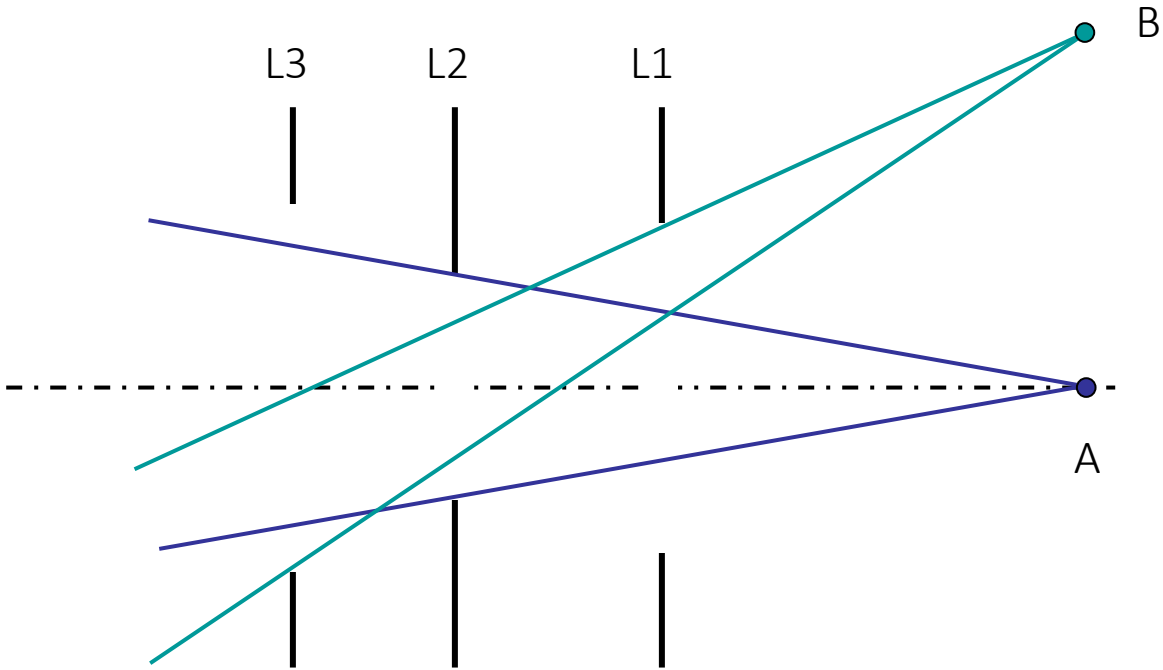
Chromatic Aberration

longitudinal chromatic aberration
(axial)



transverse chromatic aberration
(lateral)

Vignetting



More light passes through lens L3 for scene point A than scene point B

Results in spatially non-uniform brightness (in the periphery of the image)

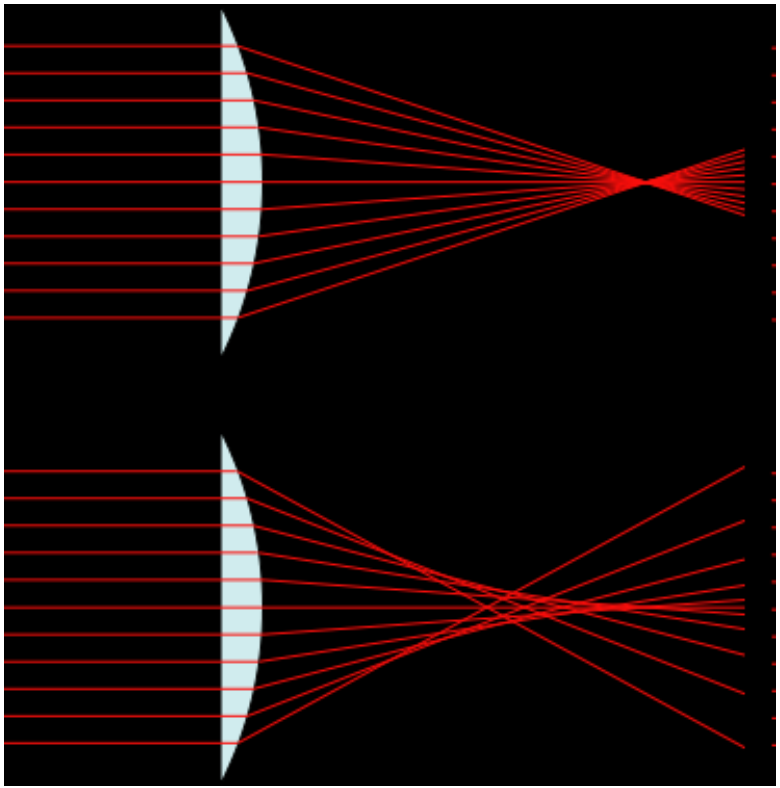
Vignetting

Images from wikipedia

KAIST

Spherical Aberration

- Spherical lenses don’t focus light perfectly.
  - Rays farther from the optical axis focus closer

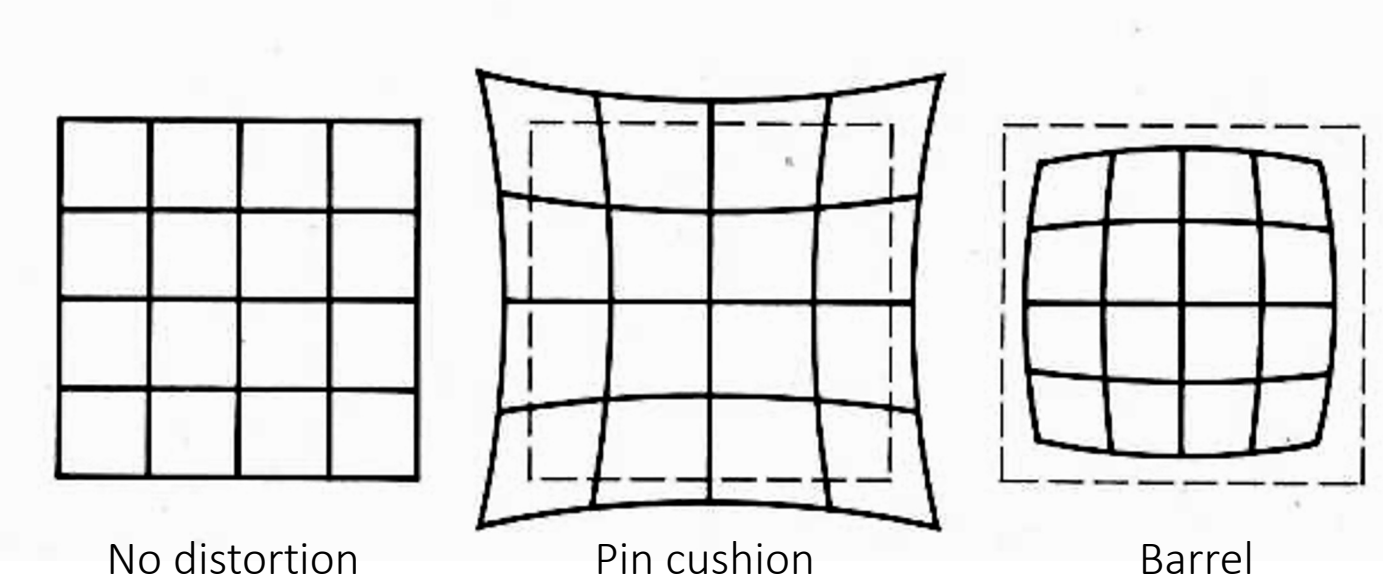


Conceptual ray diagrams of ideal and spherically aberrated lenses.

KAIST

Radial Distortion

- Caused by imperfect lenses
- Deviations are most noticeable for rays that pass through the edge of the lens



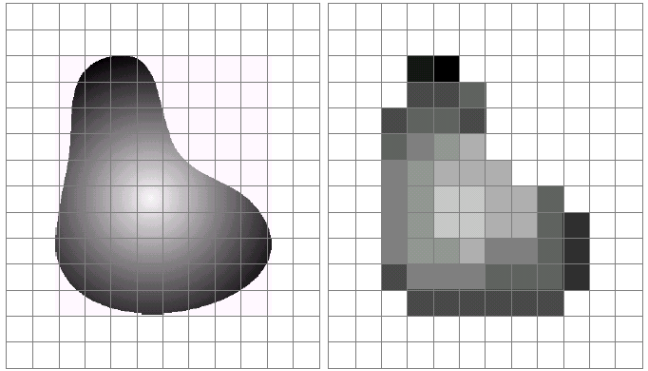
No distortion Pin cushion Barrel

Radial Distortion Correction

from Helmut Dersch

KAIST

Digital Camera



a b

FIGURE 2.17 (a) Continuos image projected onto a sensor array. (b) Result of image sampling and quantization.

- A digital camera replaces film with a sensor array
  - Each cell in the array is light-sensitive diode that converts photons to electrons
  - Two common types
    - **Charge Coupled Device** (CCD)
    - **Complementary metal oxide semiconductor** (CMOS)
      http://electronics.howstuffworks.com/digital-camera.htm

Slide by Steve Seitz

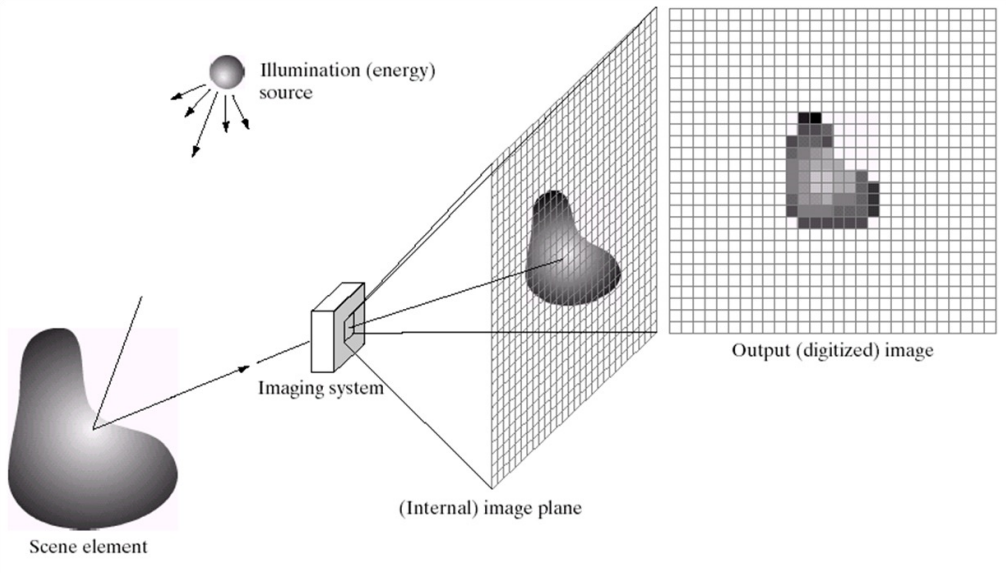
KAIST

Image Representation

- $I(x,y)$: a two-dimensional function
  - x and y are spatial coordinates.
  - The amplitude I at any pair of coordinates (x,y) is called the intensity or gray level.
  - I can be a vector representing a color image, e.g. using the RGB model, or in general a multispectral image.
  - A video signal is similarly expressed as a sequence of frames $I(x,y,t)$.

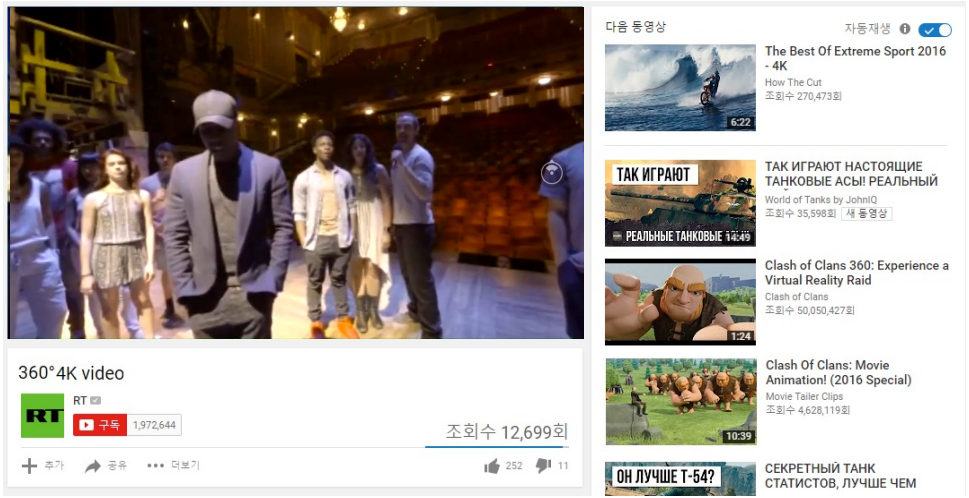
Digital Image Acquisition

- An image to a digital image
  - When x, y, and I are discrete quantities, the image is digital.
    - **Sampling** (location (x, y)) and **quantization** (gray or color values I)
    - The location in an image (x,y) is corresponding to a pixel (picture element).



New Types of Cameras

- 360 cameras
  - capture all of the surrounding area around the camera without blind spots



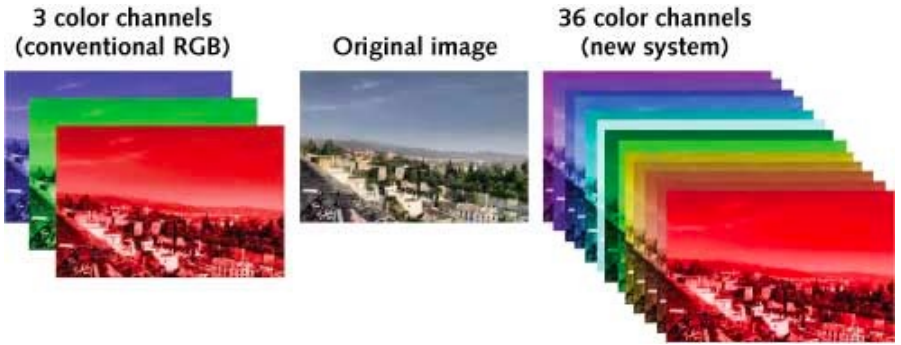
360° video player on YouTube

Samsung Gear 360 LG 360 Cam

Portable 360° cameras

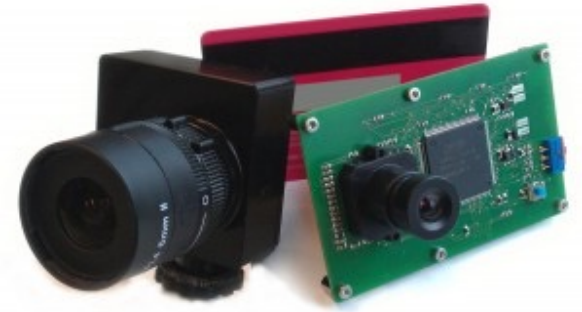
New Types of Cameras (cont.)

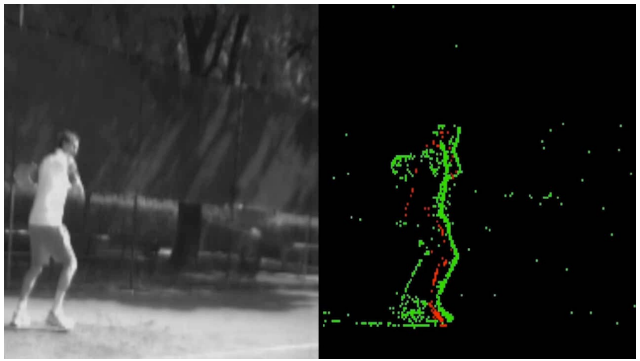
- Multi-spectral cameras
  - capture image data within specific wavelength ranges across the electromagnetic spectrum.

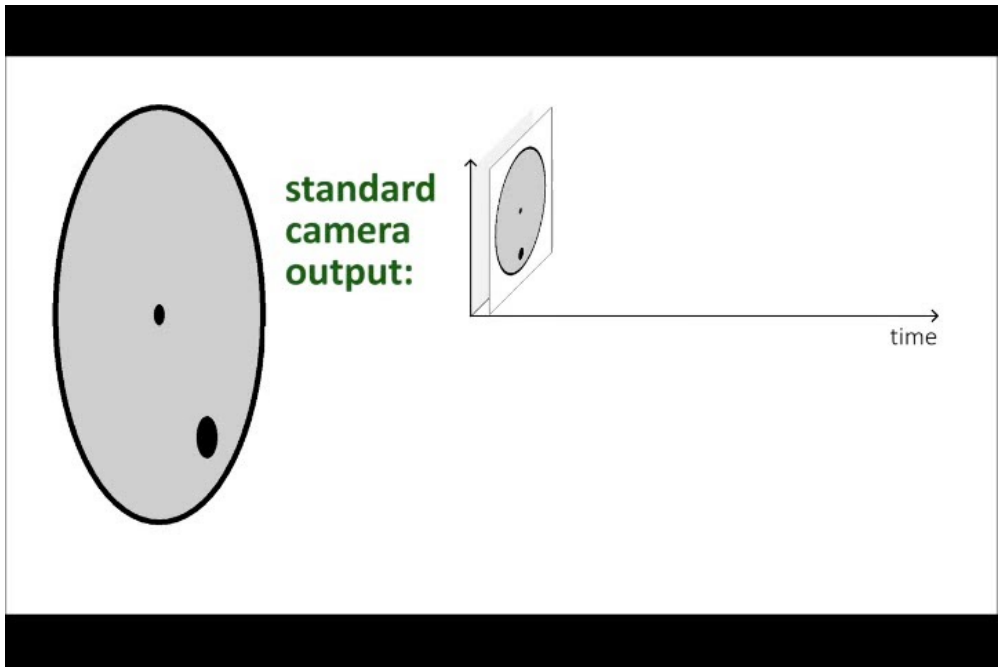


New Types of Cameras (cont.)

- Event cameras
  - bio-inspired vision sensors that output pixel-level brightness changes instead of standard intensity frames





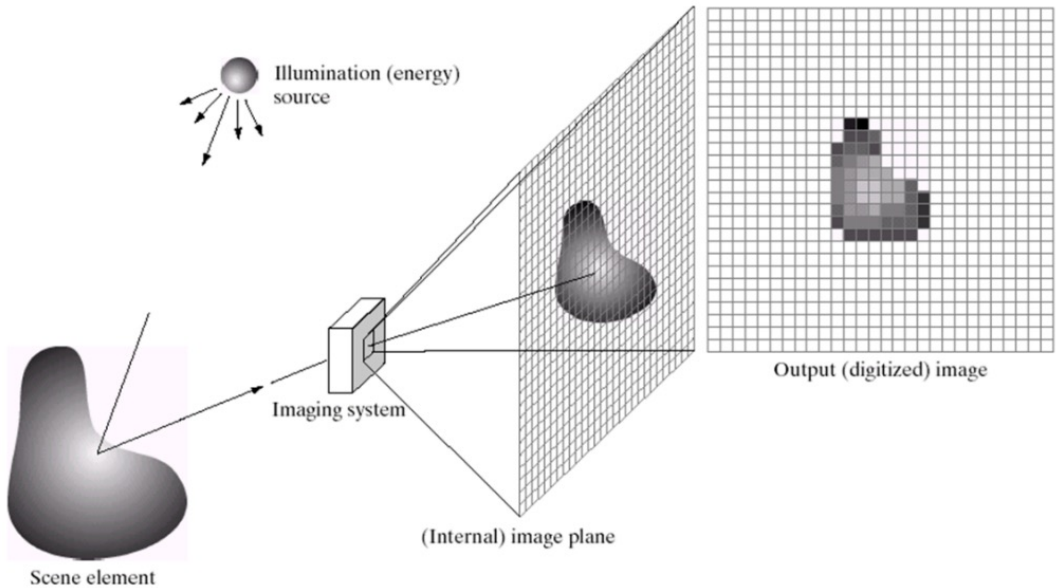


CAMERA CALIBRATION

KAIST

Geometric Vision & Photometric Vision

- Geometric vision
  - Related to how the position in an image of a scene point is determined.
- Photometric vision
  - Related to how the gray (or color) values of pixels are determined.



Geometric Vision

- The geometric computer vision is to use image data as the source of metric 3D information.
  - What cues in the image allow us to do this?
    - Visual cues related to scene geometry

Visual Cues

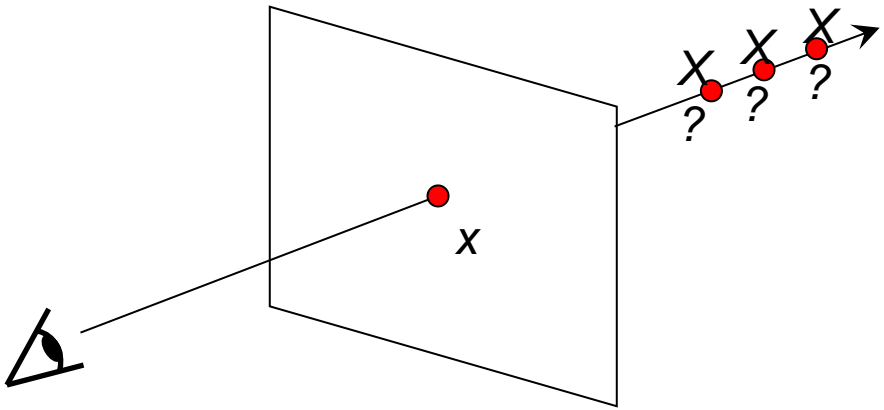
Perspective
Motion

From The Art of Photography, Canon
Merle Norman Cosmetics, Los Angeles

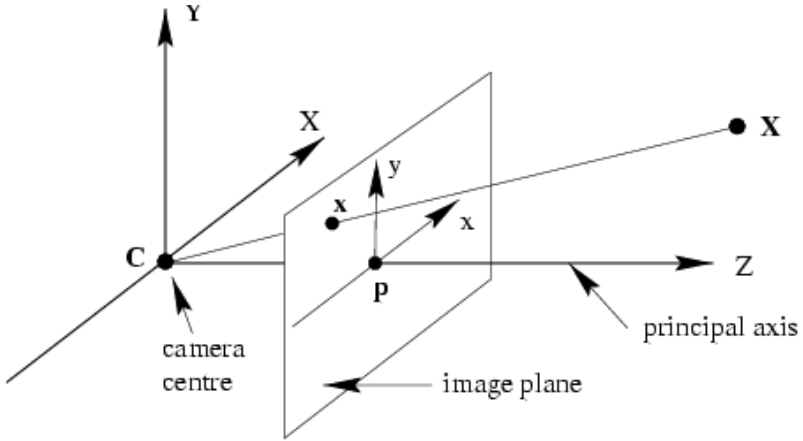
Slide credit: S. Seitz

Recovery of 3D Structure

- Focusing on perspective and motion
- Multi-view geometry required
  - The recovery of structure from one image is inherently ambiguous.

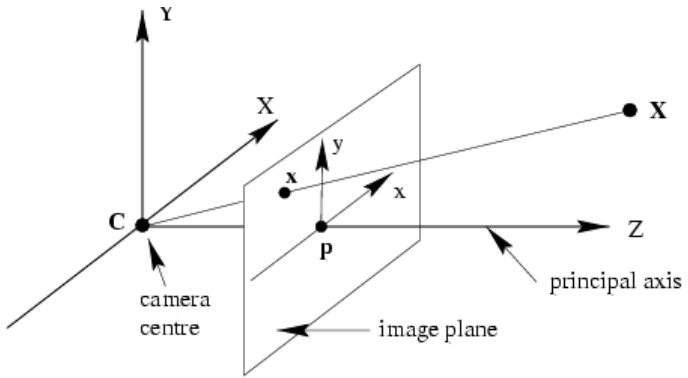


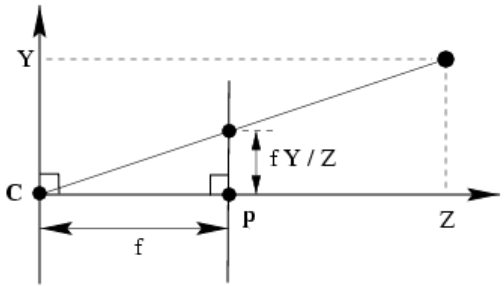
Camera Coordinate System



- Principal axis
  - Line from the camera center perpendicular to the image plane
- Normalized (camera) coordinate system
  - Camera center is at the origin and the principal axis is the z-axis.
- Principal point (p)
  - Point where principal axis intersects the image plane (origin of normalized coordinate system)

Pinhole Camera Model with Homogeneous Coordinates



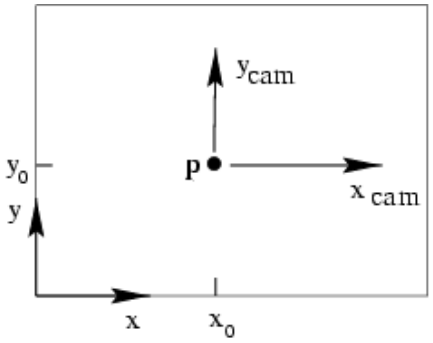


$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \sim \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} f & & \\ & f & \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

$$\mathrm{P} = \mathrm{diag}(f, f, 1)\left[\mathrm{I} \mid 0\right]$$

Principal Point Offset

- Camera coordinate system
  - Origin is at the principal point.
- Image coordinate system
  - Origin is in the corner.



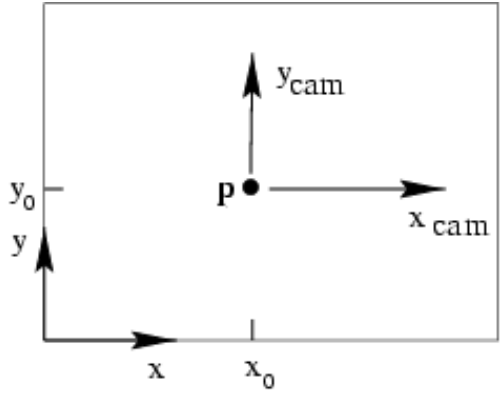
principal point: (p_x, p_y)

$$(X,Y,Z) \mapsto (fX/Z + p_x, fY/Z + p_y)$$

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX + Zp_x \\ fY + Zp_x \\ Z \end{pmatrix} = \begin{bmatrix} f & & p_x & 0 \\ & f & p_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

KAIST

Principal Point Offset



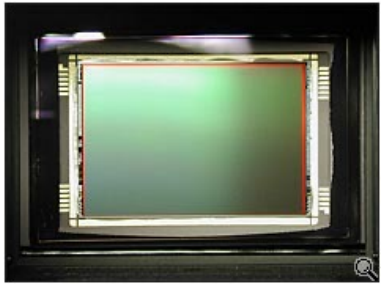
principal point: (p_x, p_y)

$$\begin{pmatrix} fX + Zp_x \\ fY + Zp_x \\ Z \end{pmatrix} = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

$$K = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}$$ calibration matrix

$$\mathbf{P} = \mathbf{K}[\mathbf{I} \mid \mathbf{0}]$$

Pixel Coordinates

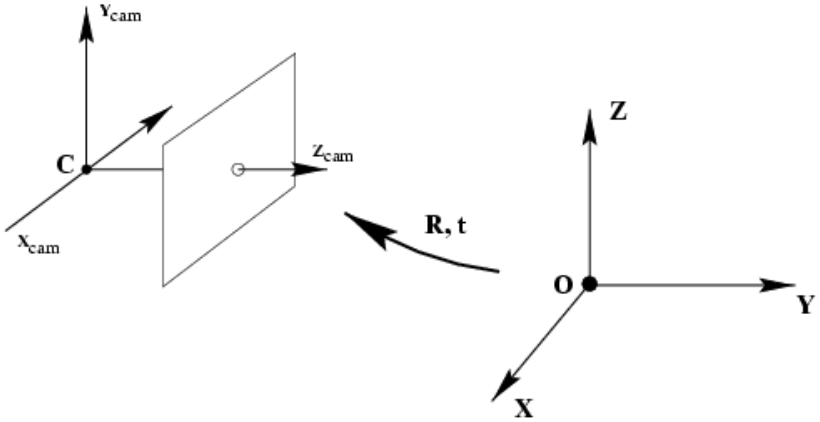


Pixel size: $\frac{1}{m_x} \times \frac{1}{m_y}$

- m_x pixels per meter in horizontal direction,
  m_y pixels per meter in vertical direction

$$K = \underbrace{\begin{bmatrix} m_x & & \\ & m_y & \\ & & 1 \end{bmatrix}}_{\text{pixels/m}} \underbrace{\begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}}_{\text{m}} = \underbrace{\begin{bmatrix} \alpha_x & & \beta_x \\ & \alpha_y & \beta_y \\ & & 1 \end{bmatrix}}_{\text{pixels}}$$

Camera Rotation and Translation



- In general, the camera coordinate frame will be related to the world coordinate frame by a rotation and a translation.

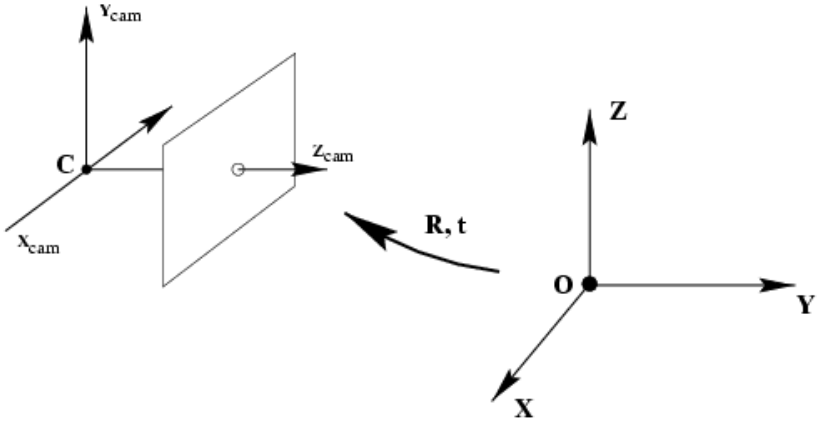
$$\tilde{X}_{cam} = R\left(\tilde{X} - \tilde{C}\right)$$

coords. of point in camera frame

coords. of a point in world frame (nonhomogeneous)

coords. of camera center in world frame

Camera Rotation and Translation



In non-homogeneous coordinates:

$$\tilde{X}_{cam} = R\left(\tilde{X} - \tilde{C}\right)$$

$$X_{cam} = \begin{bmatrix} R & -R\tilde{C} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} \tilde{X} \\ 1 \end{pmatrix} = \begin{bmatrix} R & -R\tilde{C} \\ 0 & 1 \end{bmatrix} X$$

$$x = K[I \mid 0]X_{cam} = K[R \mid -R\tilde{C}]X \qquad P = K[R \mid t], \quad t = -R\tilde{C}$$

Note: C is the null space of the camera projection matrix (PC=0)

Camera Parameters

- A camera is described by several parameters
  - Extrinsic parameters
    - translation **t** of the optical center from the origin of world cords
    - rotation **R** of the image plane
  - Intrinsic parameters
    - focal length f
    - principle point (x'_c, y'_c)
    - pixel size (s_x, s_y)

- Projection equation

$$\mathbf{x} = \begin{bmatrix} sx \\ sy \\ s \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{\Pi X} \qquad \mathbf{\Pi} = \begin{bmatrix} -fs_x & 0 & x'_c \\ 0 & -fs_y & y'_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{3x3} & \mathbf{0}_{3x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I}_{3x3} & \mathbf{T}_{3x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix}$$

Camera Calibration

- Given n points with known 3D coordinates X_i and known image projections x_i, estimate the camera parameters

- Why Calibrate?
  - To solve for 3D geometry
  - Alternative approach
    - Solve for 3D shape without known cameras
      - Structure from motion (unknown extrinsics)
      - Self calibration (unknown intrinsics & extrinsics)
  - Why bother pre-calibrating the camera?
    - Simplifies the 3D reconstruction problem
      - fewer parameters to solve for later on
    - Improves accuracy
    - Not too hard to do
    - Eliminates certain ambiguities (scale of scene)

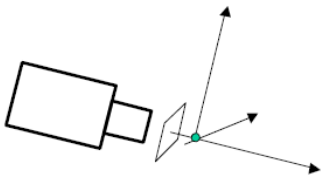
KAIST

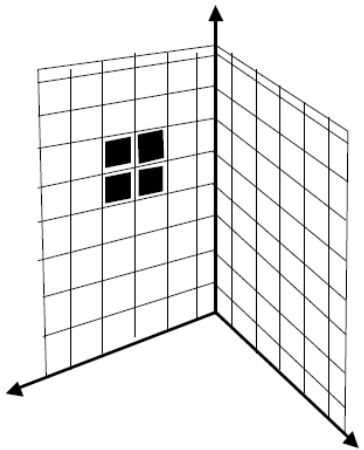
Camera Calibration

- Given *n* points with known 3D coordinates X_i and known image projections x_i, estimate the camera parameters

- Procedure
  - Estimating the matrix **P** using scene points and their images using known patterns.
  - Estimating the intrinsic parameters and the extrinsic parameters from **P**

$$\mathbf{P} = \mathbf{K}\mathbf{R}\,[\mathbf{I}_3 \mid -\tilde{\mathbf{C}}]$$





Computing Projection Matrix

- $\mathbf{P} = \mathbf{KR}\,[\mathbf{I}_3 \mid -\tilde{\mathbf{C}}]$
  - Left 3x3 submatrix of **P** is the product of upper-triangular matrix and orthogonal matrix.
  - Left 3x3 submatrix **KR** is non-singular.

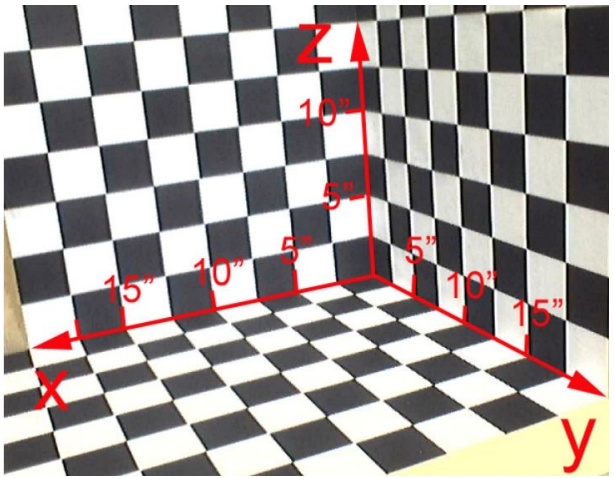
$$\mathbf{x}_i = \mathbf{P}\mathbf{X}_i \Rightarrow [\mathbf{x}_i]_\times \mathbf{P}\mathbf{X}_i = \mathbf{0}$$

$$\begin{bmatrix} \mathbf{0}^\top & -w_i\mathbf{X}_i^\top & y_i\mathbf{X}_i^\top \\ w_i\mathbf{X}_i^\top & \mathbf{0}^\top & -x_i\mathbf{X}_i^\top \\ -y_i\mathbf{X}_i^\top & x_i\mathbf{X}_i^\top & \mathbf{0}^\top \end{bmatrix} \begin{pmatrix} \mathbf{P}^1 \\ \mathbf{P}^2 \\ \mathbf{P}^3 \end{pmatrix} = \mathbf{0}$$

two linearly independent equations

$$\begin{bmatrix} \mathbf{0}^\top & -w_i\mathbf{X}_i^\top & y_i\mathbf{X}_i^\top \\ w_i\mathbf{X}_i^\top & \mathbf{0}^\top & -x_i\mathbf{X}_i^\top \end{bmatrix} \begin{pmatrix} \mathbf{P}^1 \\ \mathbf{P}^2 \\ \mathbf{P}^3 \end{pmatrix} = \mathbf{0}$$

For n points, we have Ap=0 where A: 2n x 12 matrix and p: 12x1 vector (p^i is a row vector of P)



cross product of two vectors: $\mathbf{x} \times \mathbf{y} = [\mathbf{x}]_\times \mathbf{y}$

$$[\mathbf{x}]_\times = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}_\times = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}$$

(skew symmetric matrix s.t. $[\mathbf{x}]_\times = -[\mathbf{x}]_\times^T$)

Algorithm step 1: Compute the matrix P

$$\mathbf{x}_i = \mathtt{P}\mathbf{X}_i.$$

Each correspondence generates two equations

$$x_i = \frac{p_{11}\mathrm{X}_i + p_{12}\mathrm{Y}_i + p_{13}\mathrm{Z}_i + p_{14}}{p_{31}\mathrm{X}_i + p_{32}\mathrm{Y}_i + p_{33}\mathrm{Z}_i + p_{34}} \qquad y_i = \frac{p_{21}\mathrm{X}_i + p_{22}\mathrm{Y}_i + p_{23}\mathrm{Z}_i + p_{24}}{p_{31}\mathrm{X}_i + p_{32}\mathrm{Y}_i + p_{33}\mathrm{Z}_i + p_{34}}$$

Multiplying out gives equations linear in the matrix elements of P

$$x_i(p_{31}\mathrm{X}_i + p_{32}\mathrm{Y}_i + p_{33}\mathrm{Z}_i + p_{34}) = p_{11}\mathrm{X}_i + p_{12}\mathrm{Y}_i + p_{13}\mathrm{Z}_i + p_{14}$$
$$y_i(p_{31}\mathrm{X}_i + p_{32}\mathrm{Y}_i + p_{33}\mathrm{Z}_i + p_{34}) = p_{21}\mathrm{X}_i + p_{22}\mathrm{Y}_i + p_{23}\mathrm{Z}_i + p_{24}$$

These equations can be rearranged as

$$\begin{pmatrix} \mathrm{X} & \mathrm{Y} & \mathrm{Z} & 1 & 0 & 0 & 0 & 0 & -x\mathrm{X} & -x\mathrm{Y} & -x\mathrm{Z} & -x \\ 0 & 0 & 0 & 0 & \mathrm{X} & \mathrm{Y} & \mathrm{Z} & 1 & -y\mathrm{X} & -y\mathrm{Y} & -y\mathrm{Z} & -y \end{pmatrix} \mathbf{p} = \mathbf{0}$$

with $\mathbf{p} = (p_{11}, p_{12}, p_{13}, p_{14}, p_{21}, p_{22}, p_{23}, p_{24}, p_{31}, p_{32}, p_{33}, p_{34})^\top$ a 12-vector.

Computing Projection Matrix

- Minimal solution
  - $\mathbf{p}$ has 11 dof (5 from intrinsic parameters, 3 from $\mathbf{R}$, 3 from $\mathbf{C}$)
  - 2 independent eq./point.
    - 5½ correspondences needed (say 6)

- Over-determined solution
  - $n \geq 6$ points
    - When possible, have at least 5 times as many equations as unknowns (28 points)
  - minimize $\|\mathbf{Ap}\|$ subject to constraint $\|\mathbf{p}\|=1$
    - $\mathbf{p}$ is the unit singular vector of $\mathbf{A}$ corresponding to the smallest singular value (the last column of $\mathbf{V}$, where $\mathbf{A} = \mathbf{UDV}^T$ is the SVD of $\mathbf{A}$)
    - Called Direct Linear Transformation (DLT)

Computing Projection Matrix

- Improving **P** solution with non-linear minimization
  - Finding **P** using DLT (1)
  - Using the result of (1) as an initialization and performing non-linear minimization of

$$\sum_i d(\mathbf{x}_i, \mathbf{P}\mathbf{X}_i)^2$$

with Levenberg-Marquardt iterative minimization

Computing Projection Matrix

- Improving **P** solution with non-linear minimization

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \sim \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & 1 \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

  - Feature measurement equations

$$u_i = \frac{m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1}$$

$$v_i = \frac{m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1}$$

Minimize "image-space error"

$$e(\mathrm{M}) = \sum_i [(u_i - \frac{m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1})^2 + (v_i - \frac{m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1})^2]$$

KAIST

Intrinsic and Extrinsic Parameters

- Once we've recovered the numerical form of the camera matrix, we still have to figure out the intrinsic and extrinsic parameters.

- This is a matrix decomposition problem, not an estimation problem.

- Procedure
  - Finding homogeneous coordinates of **C** in the scene from **P**
  - Finding camera orientation and intrinsic parameters

Intrinsic and Extrinsic Parameters

$$\mathbf{P} = \mathbf{KR}\,[\mathbf{I}_3 \mid -\tilde{\mathbf{C}}]$$

- Step 1.
  - $\mathbf{C} = [\tilde{\mathbf{C}}\ \ \mathbf{1}]^{\mathrm{T}}$ is the null vector of matrix P (PC=0)
  - Find the null vector C of P using SVD
    - C is the unit singular vector of P corresponding to the smallest singular value (the last column of V, where $P = UDV^T$ is the SVD of P)

- Step 2.
  - Left 3x3 submatrix M of P is of form M=KR
    - K is an upper triangular matrix and R is an orthogonal matrix
  - Any non-singular square matrix M can be decomposed into the product of an upper-triangular matrix K and an orthogonal matrix R using the RQ factorization
    - Similar to QR factorization but order of 2 matrices is reversed.

KAIST

Algorithm step 2: Decompose P into K, R and t

The first 3×3 submatrix, M, of P is the product ($\mathtt{M} = \mathtt{KR}$) of an upper triangular and rotation matrix.

(i) Factor M into KR using the QR matrix decomposition. This determines K and R.

(ii) Then

$$\mathbf{t} = \mathtt{K}^{-1}(p_{14}, p_{24}, p_{34})^\top$$

Note, this produces a matrix with an extra skew parameter s

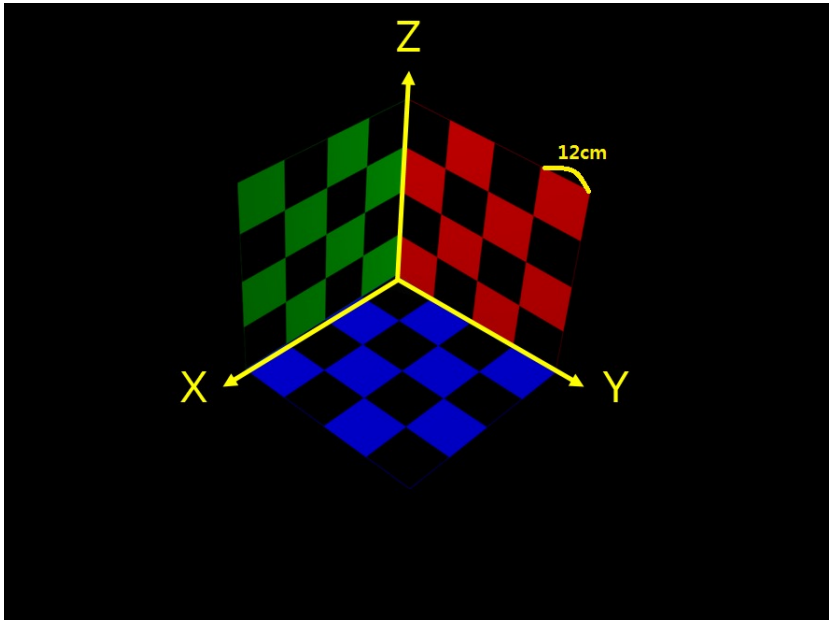
$$\mathtt{K} = \begin{bmatrix} \alpha_x & s & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix}$$

with $s = \tan\theta$, and θ the angle between the image axes.

Camera Calibration

Calibrate a virtual camera that captures a 3D calibration rig as shown in Figure 1 below. The rig is placed at the origin of the world frame. In addition, planes for the rig are orthogonal, and the axis of the world frame and the lines of intersections of planes are shared. The width and height of each square for the check board is set to be 12cm.

From these settings, estimate an intrinsic matrix (K), a rotation matrix (R), and a translation vector (t). For coordinates of corresponding 2D points to 3D points, you could gather the position of each points by hand. To check sanity of your code, compare the estimated position of a camera center in the world frame with ground truth (X= 166.20, Y=141.46, Z=170.08).



KAIST

```
clear all
% pixels
image_points = [
      532      760       1
      228      707       1
      543      918       1
      826      779       1
      353      453       1
      946      786       1
      370     1136       1
      453      625       1
      718      772       1
      460      928       1
      545      461       1
      825      620       1
      826      936       1
      555     1100       1
      ]';
% 3D points
world_points = [
      0       0       0   1.0000
  0.1200      0   0.4800   1.0000
      0   0.2400   0.1200   1.0000
  0.3600   0.3600       0   1.0000
  0.4800       0   0.4800   1.0000
  0.4800   0.4800       0   1.0000
      0   0.4800   0.4800   1.0000
  0.2400       0   0.2400   1.0000
  0.2400   0.2400       0   1.0000
      0   0.2400   0.2400   1.0000
  0.4800       0   0.2400   1.0000
  0.4800   0.2400       0   1.0000
  0.2400   0.4800       0   1.0000
      0   0.4800   0.2400   1.0000
      ]';

 [d_p, n_p] = size(image_points);
```

```
% image data normalization
mean_image_points = mean(image_points');
dist_sum=0;
for i=1:n_p
   dist_sum=dist_sum+norm(image_points(:, i)-
mean_image_points')^2;
end
scale_2 = sqrt(dist_sum/2/n_p);

T_image = [1/scale_2, 0, -
1/scale_2*mean_image_points(1);
   0, 1/scale_2, -1/scale_2*mean_image_points(2);
   0, 0, 1];
norm_image_points = T_image*image_points;


% world data normalization
mean_world_points = mean(world_points');
dist_sum=0;
for i=1:n_p
   dist_sum=dist_sum+norm(world_points(:, i)-
mean_world_points')^2;
end
scale_3 = sqrt(dist_sum/3/n_p);

T_world = [1/scale_3, 0, 0, -
1/scale_3*mean_world_points(1);
   0, 1/scale_3, 0, -1/scale_3*mean_world_points(2);
   0, 0, 1/scale_3, -1/scale_3*mean_world_points(2);
   0, 0, 0, 1];
norm_world_points = T_world*world_points;
```

```
% Data matrix construction
A=zeros(2*n_p, 12);
for i=1:n_p
   A((i-1)*2+1, :) = [norm_world_points(1,i), norm_world_points(2, i),
norm_world_points(3, i), 1, 0, 0, 0, 0, -norm_image_points(1,
i)*norm_world_points(1, i), -norm_image_points(1,
i)*norm_world_points(2, i), -norm_image_points(1,
i)*norm_world_points(3, i), -norm_image_points(1, i)];
   A((i-1)*2+2, :) = [0, 0, 0, 0, -norm_world_points(1, i), -
norm_world_points(2, i), -norm_world_points(3, i), -1,
norm_image_points(2, i)*norm_world_points(1, i),
norm_image_points(2, i)*norm_world_points(2, i),
norm_image_points(2, i)*norm_world_points(3, i),
norm_image_points(2, i)];
End
% projection matrix computation
[U, S, V] = svd(A);
p = V(:, 12);
proj = reshape(p, 4, 3)';

% backward transform to fit to the original data
proj_ori = inv(T_image)*proj*T_world;

% camera center position in the world frame
C = null(proj_ori);
Co = C/C(4);

% QR factorization
M = proj_ori(1:3, 1:3);
[R_inv K_inv] =qr(inv(M));
R=inv(R_inv);
% K matrix normalization (making the (3,3) element 1)
K_temp=inv(K_inv);
K=K_temp/K_temp(3,3);
% also rescaling the projection matrix
proj_final = proj_ori/K_temp(3,3);
t = inv(K) * proj_final(:, 4);
```