University of Zurich[UZH]

ROBOTICS & PERCEPTION GROUP

Vision Algorithms for Mobile Robotics

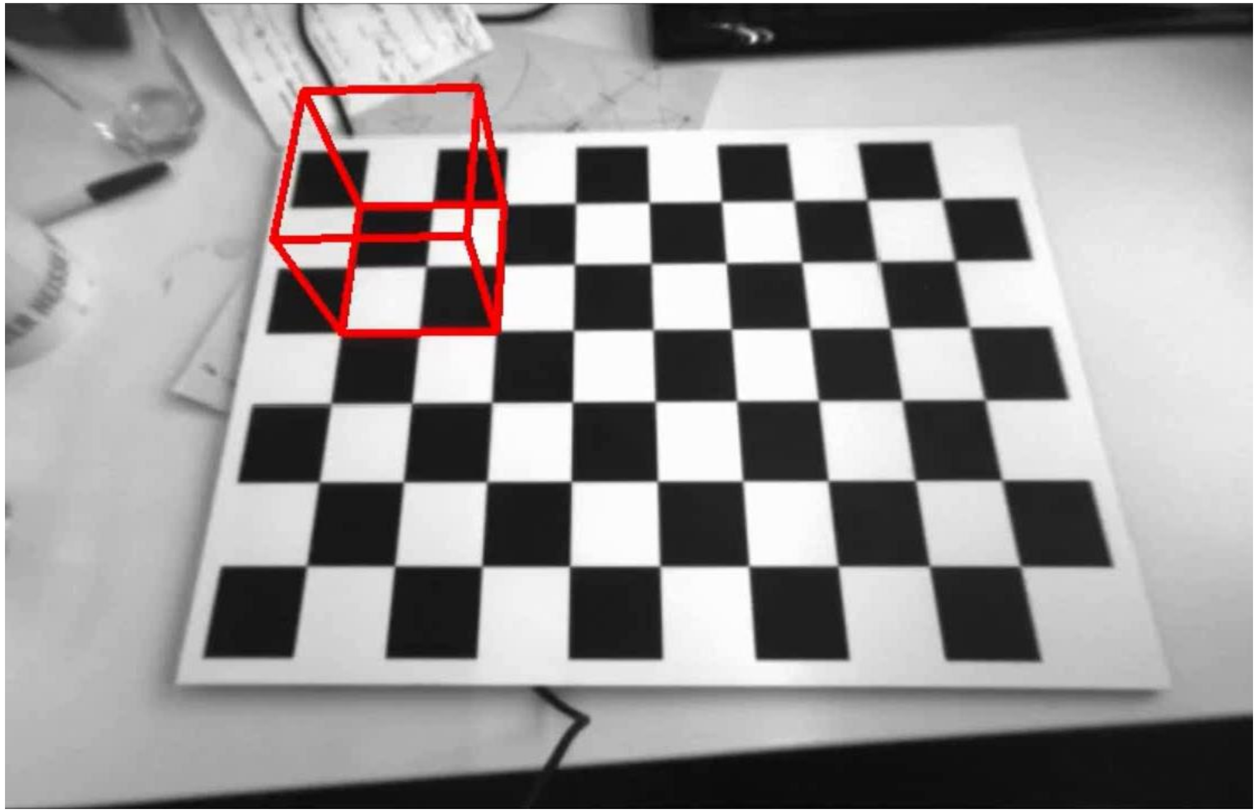
Lecture 02
Image Formation

Davide Scaramuzza

http://rpg.ifi.uzh.ch

Lab Exercise 1 - Today

Implement an augmented reality wireframe cube and practice the perspective projection

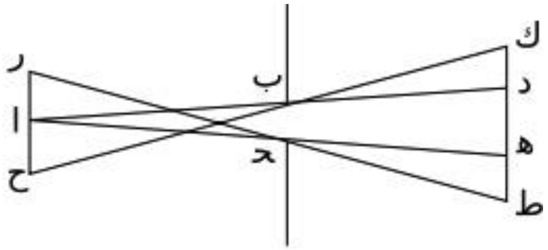


Today’s Outline

- Image Formation
- Other camera parameters
- Digital camera
- Perspective camera model
- Lens distortion

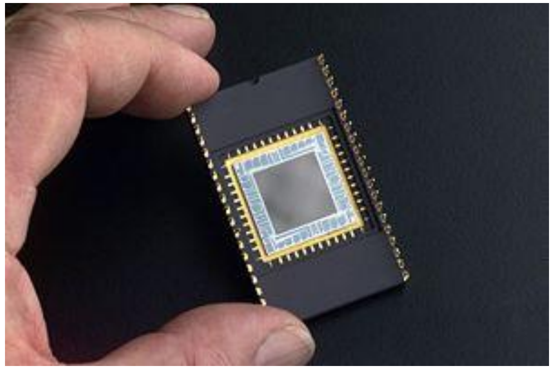
Historical context

- **Pinhole model**: Mozi (470-390 BCE), Aristotle (384-322 BCE)
- Principles of optics (including lenses):
  Alhacen (965-1039)
- **Camera obscura**: Leonardo da Vinci
  (1452-1519), Johann Zahn (1631-1707)
- **First photo (heliography)**: Joseph Nicephore Niepce (1827)
- **Daguerréotypes** (1839)
- Photographic film (Eastman, 1888, founder of Kodak)
- Cinema (Lumière Brothers, 1895)
- Color Photography (Lumière Brothers, 1908)
- Television (Baird, Farnsworth, Zworykin, 1920s)
- **First consumer camera with CCD**:
  Sony Mavica (1981)
- **First fully digital camera**: Kodak DCS100 (1990)



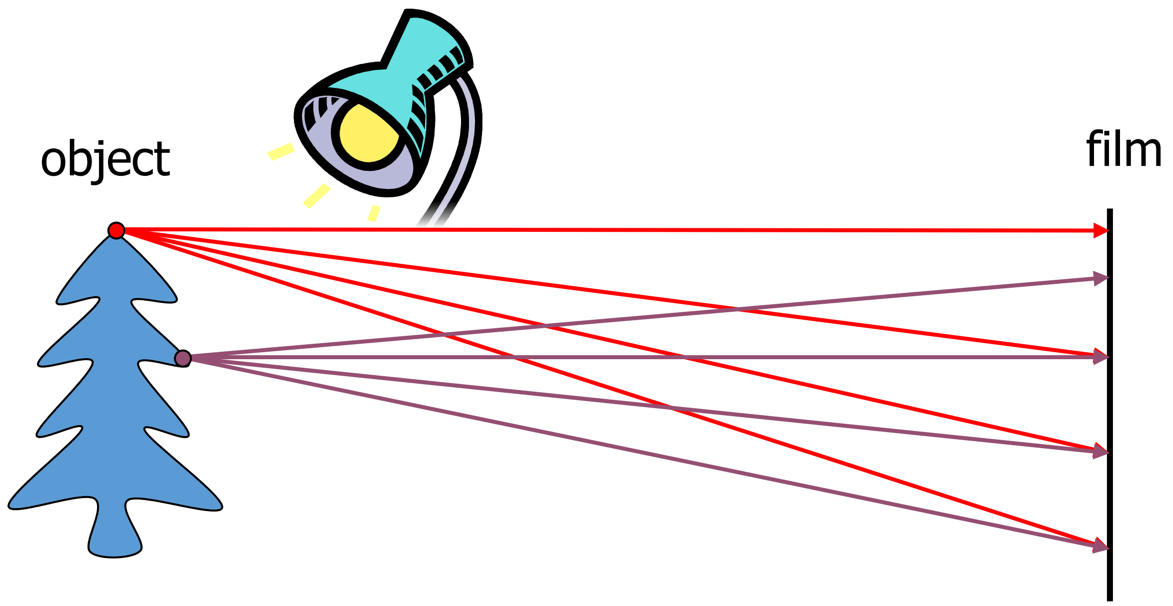
Alhacen's notes


First surviving photo ever: "View from the Window at Le Gras", by Niepce, 1827


CCD chip

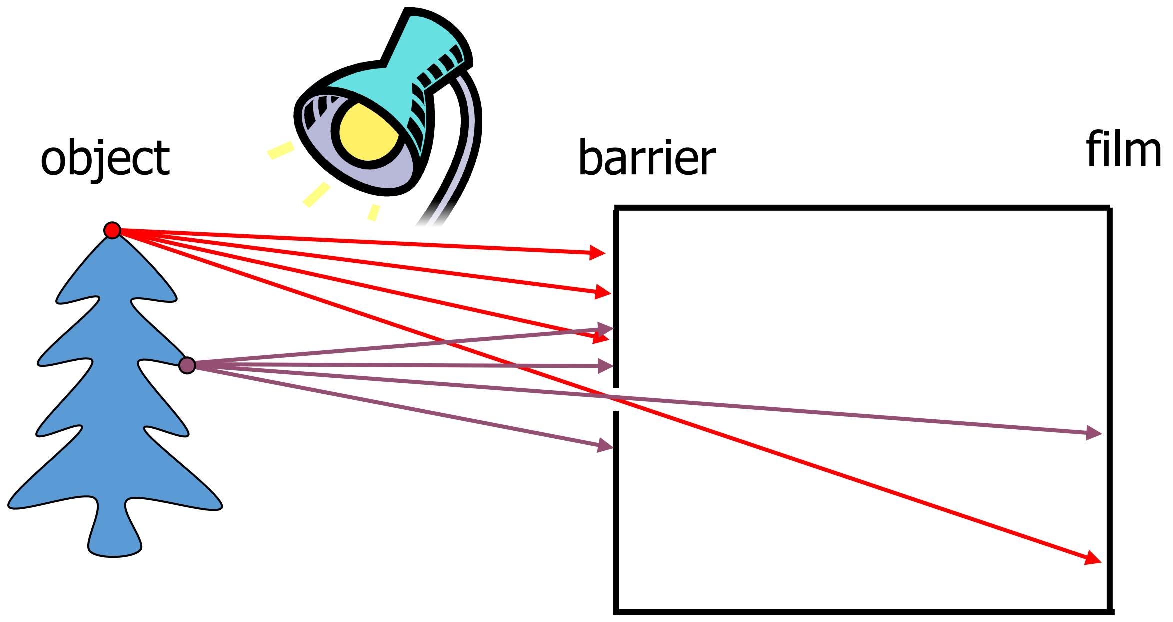
How to form an image

- Would this work?



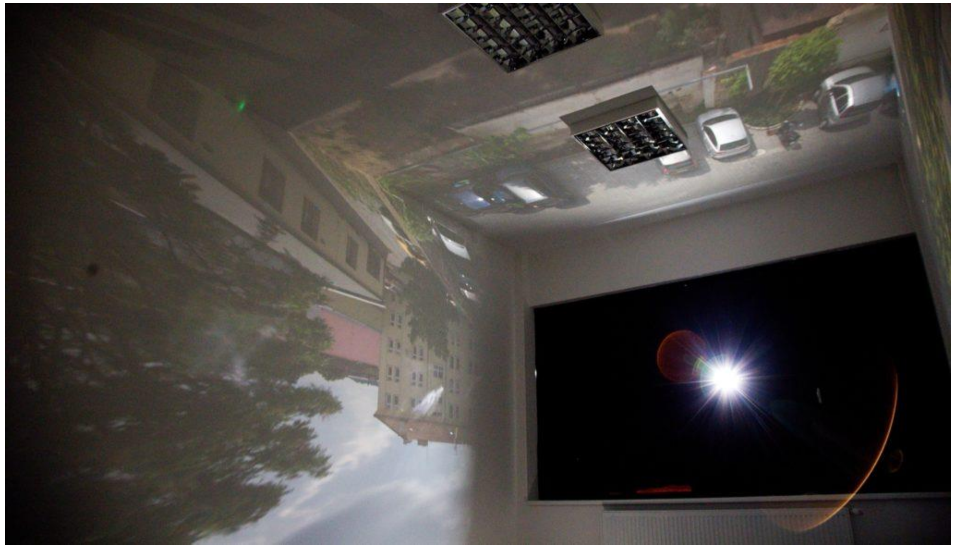
Camera obscura

- Add a barrier to block off most of the rays
  - This reduces blurring
  - The opening is known as the **aperture**



Home-made camera obscura

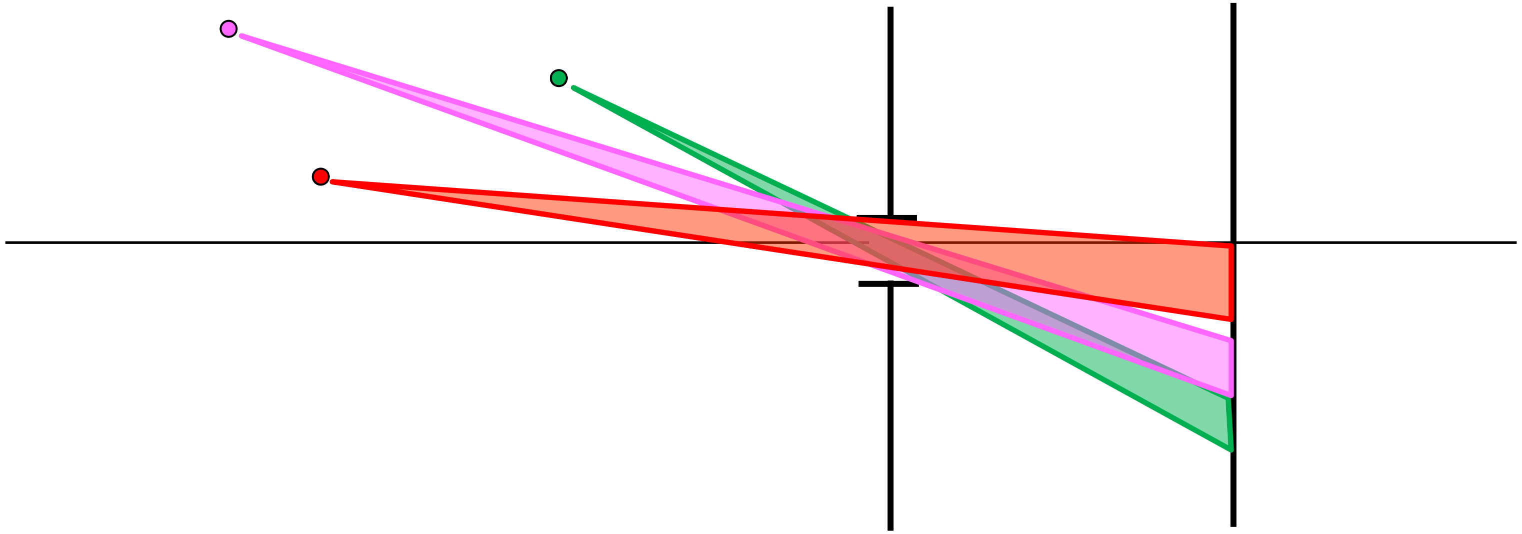
- Image is **inverted**
- **Depth of the room** is the **focal length**
- How can we **reduce the blur**?



How to build a camera obscura at home (link)

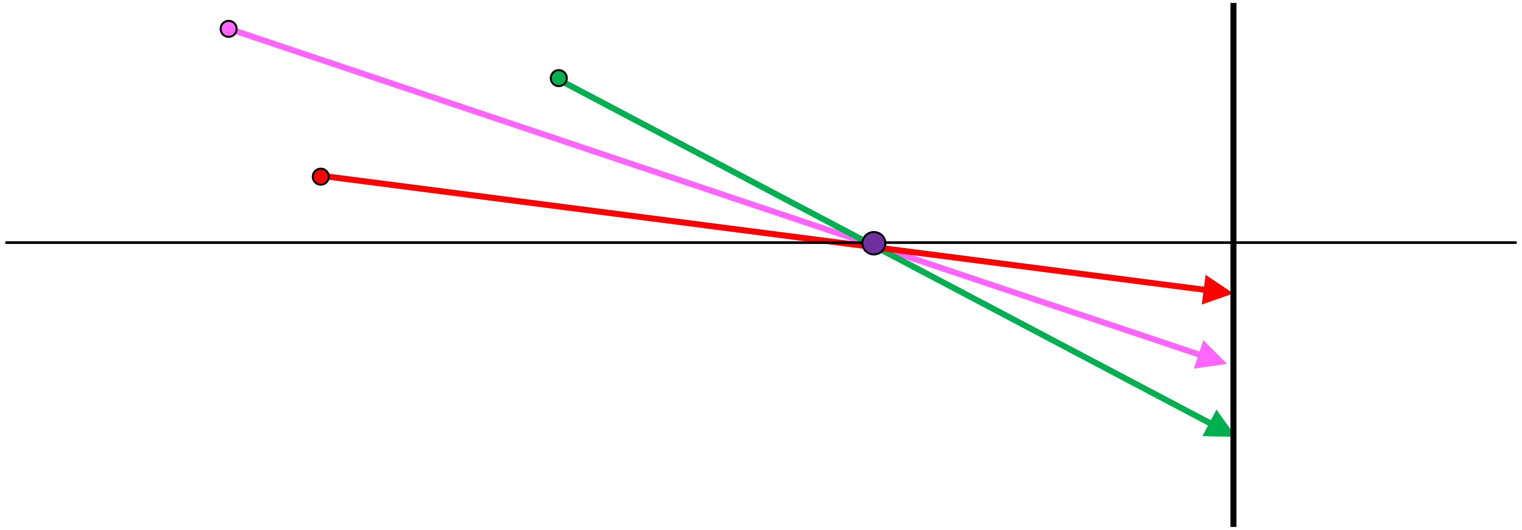
Effects of the Aperture Size

- A large **aperture** makes the **image blurry** because a cone of light is let through from each world point



Effects of the Aperture Size

- **Shrinking the** aperture makes the **image sharper**
- The ideal aperture is a **pinhole** that only lets through one ray of light from each world point



Why not making the aperture as small as possible?

- With **small apertures, less light gets through** → must **increase exposure** time
- If aperture gets **too small**, **diffraction effects** start to appear

Image formation using a converging lens

- A **thin converging lens** focuses light onto the film satisfying two properties:

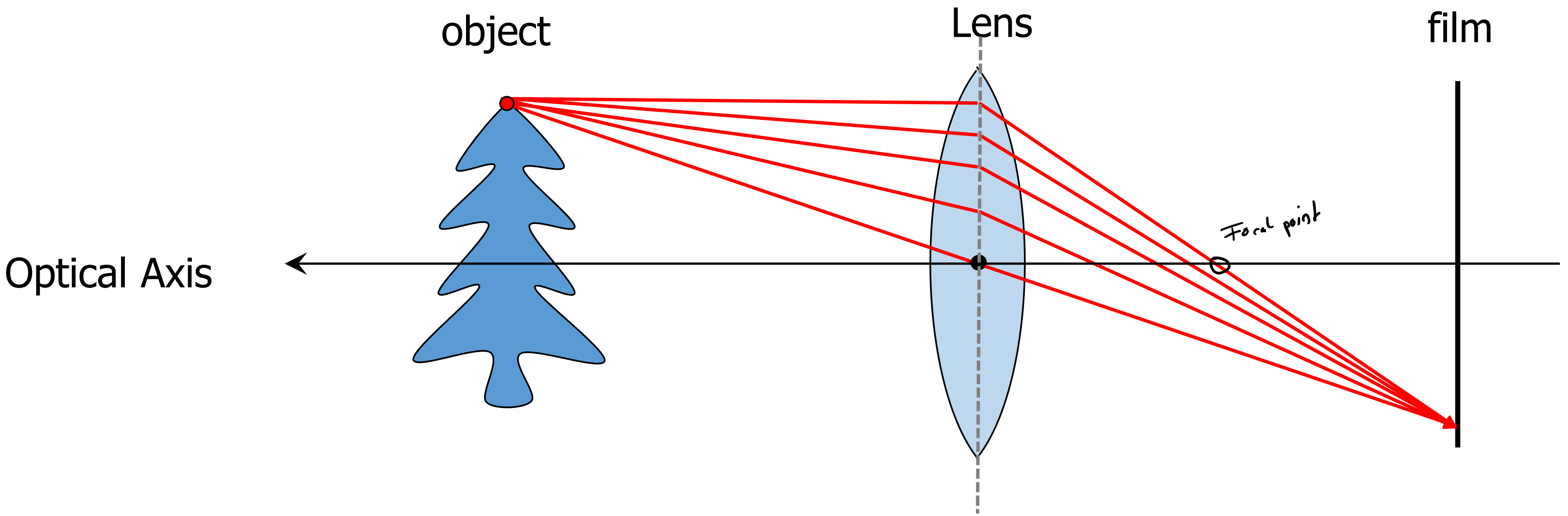


Image formation using a converging lens

- A **thin converging lens** focuses light onto the film satisfying two properties:
  1. Rays passing through the **Optical Center** are not deviated

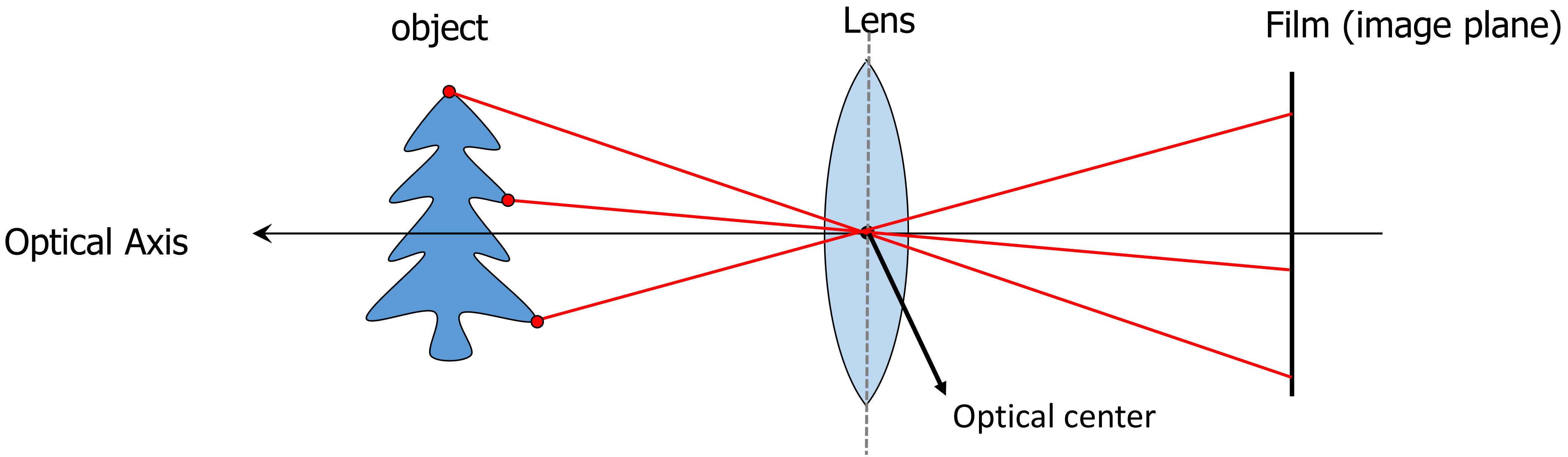
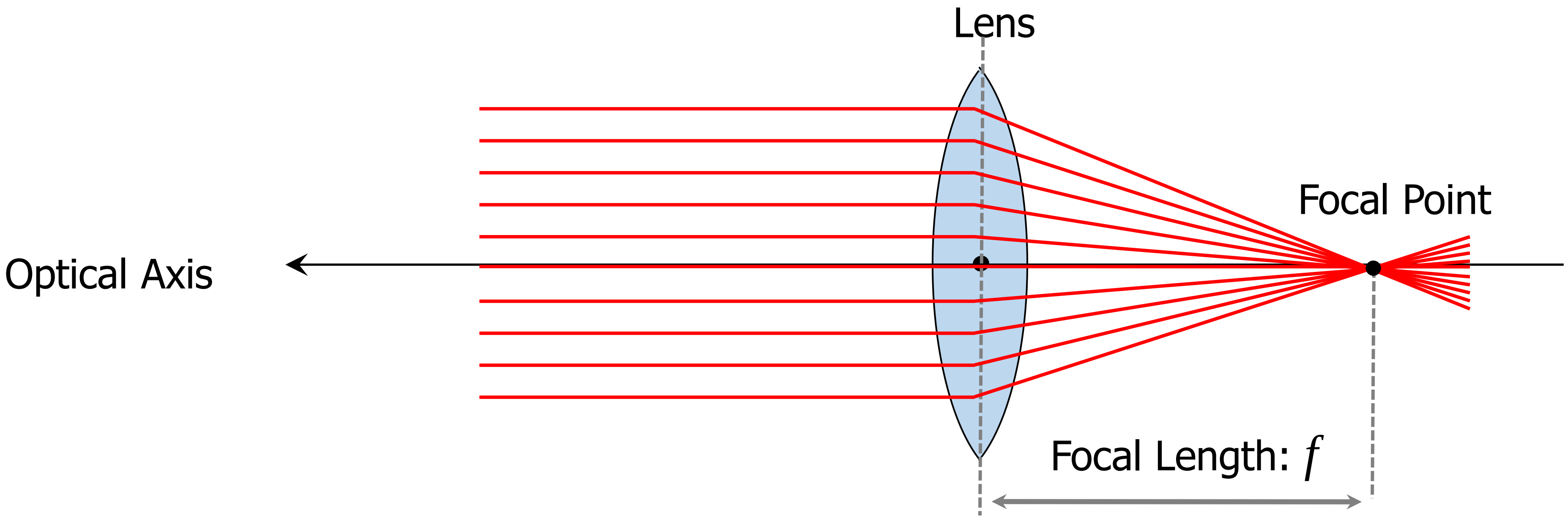


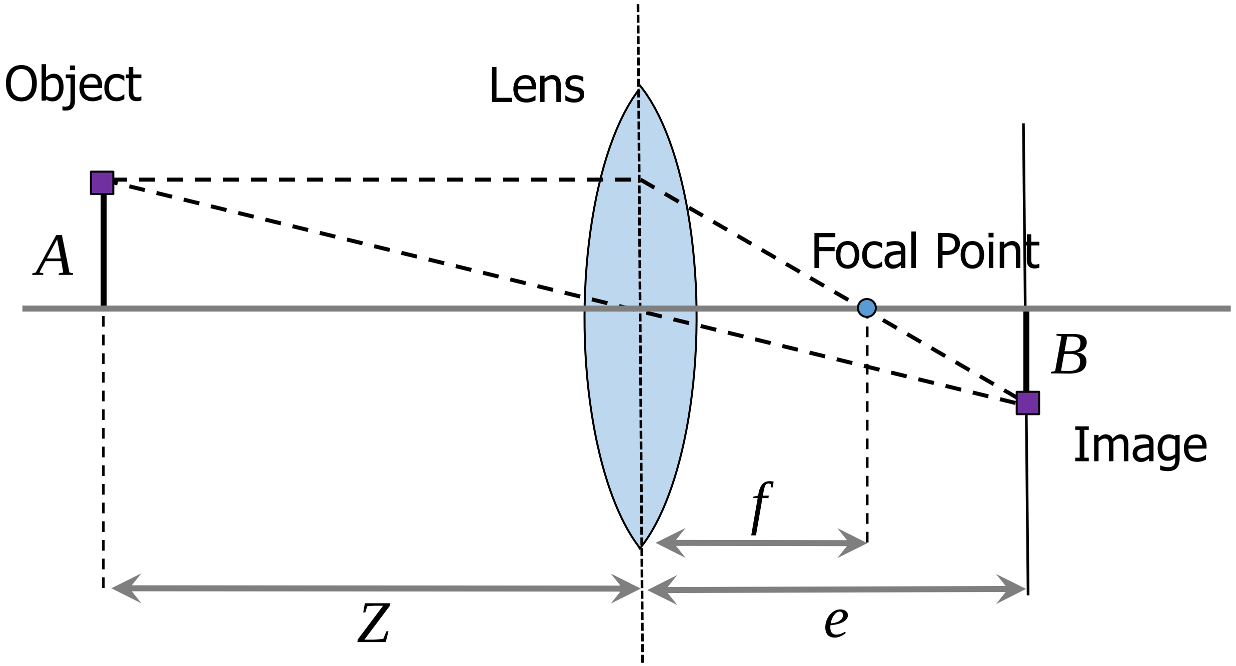
Image formation using a converging lens

- A **thin converging lens** focuses light onto the film satisfying two properties:
  1. Rays passing through the **Optical Center** are not deviated
  2. All rays parallel to the **Optical Axis** converge at the **Focal Point**



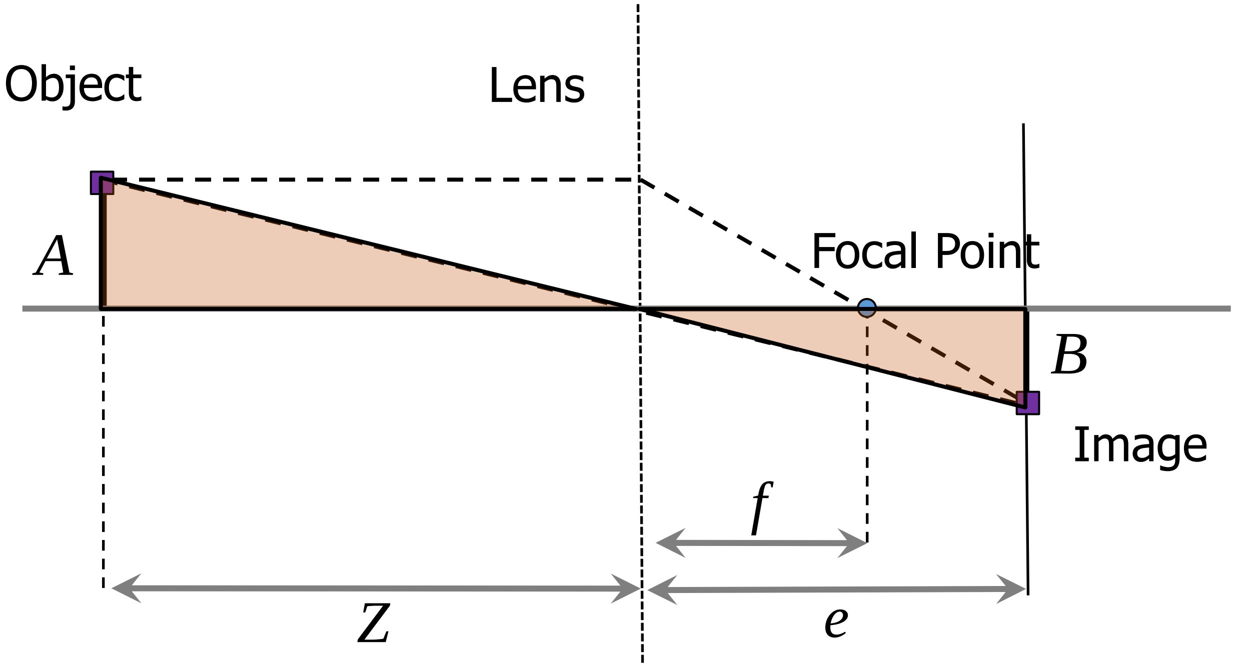
Thin lens equation

- Find a relationship between f, Z, and e



Thin lens equation

- Find a relationship between f, Z, and e

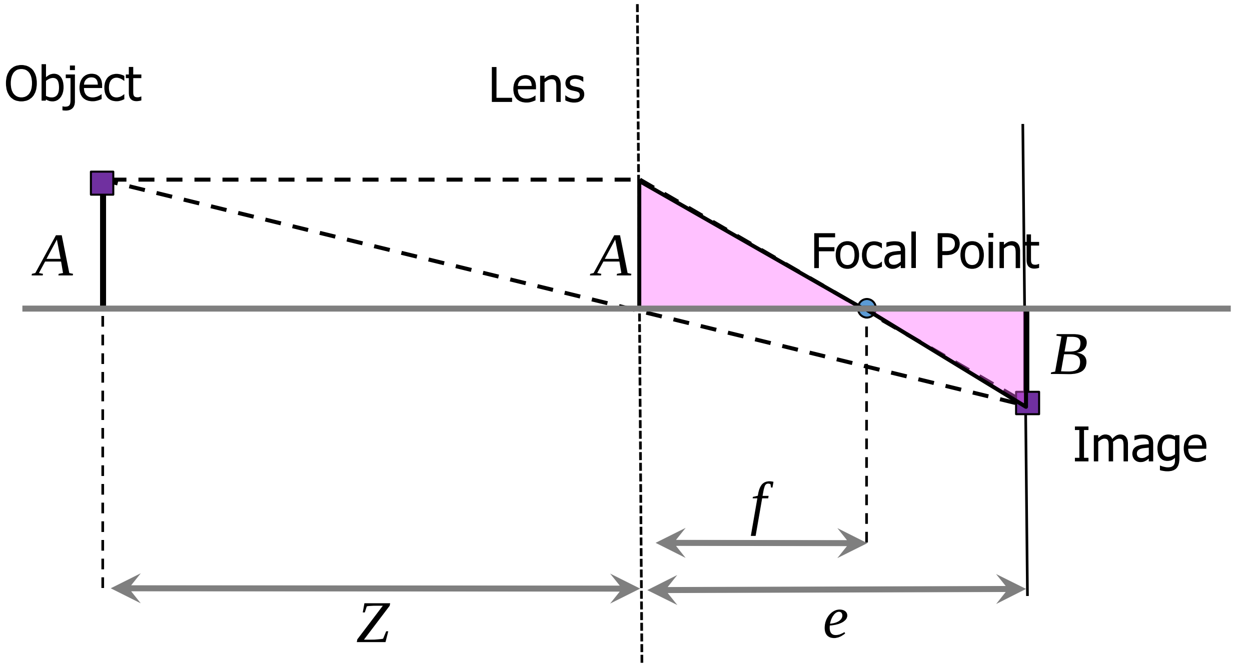


Similar Triangles:

$$\frac{B}{A} = \frac{e}{Z}$$

Thin lens equation

- Find a relationship between f, Z, and e



Similar Triangles:

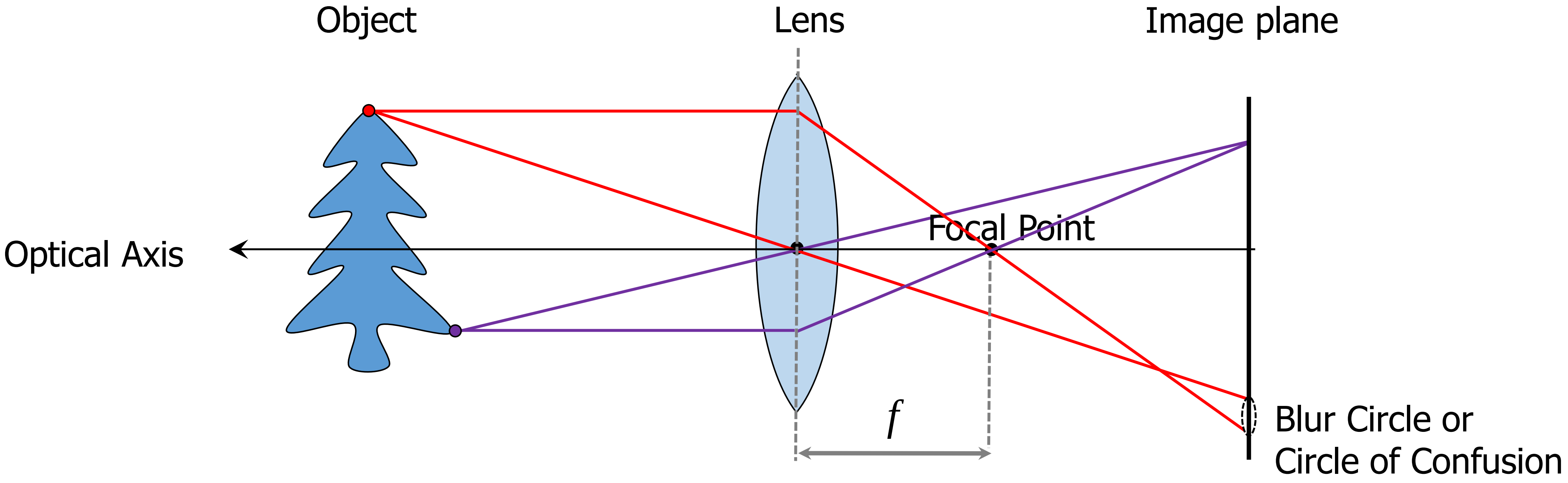
$$\left.\begin{array}{l}\dfrac{B}{A}=\dfrac{e}{Z} \\[2ex] \dfrac{B}{A}=\dfrac{e-f}{f}=\dfrac{e}{f}-1\end{array}\right\}\dfrac{e}{f}-1=\dfrac{e}{Z}\Rightarrow \boxed{\dfrac{1}{f}=\dfrac{1}{Z}+\dfrac{1}{e}}$$

"Thin lens equation"

- Any object point satisfying this equation is in focus
- Can I use this to measure distances?

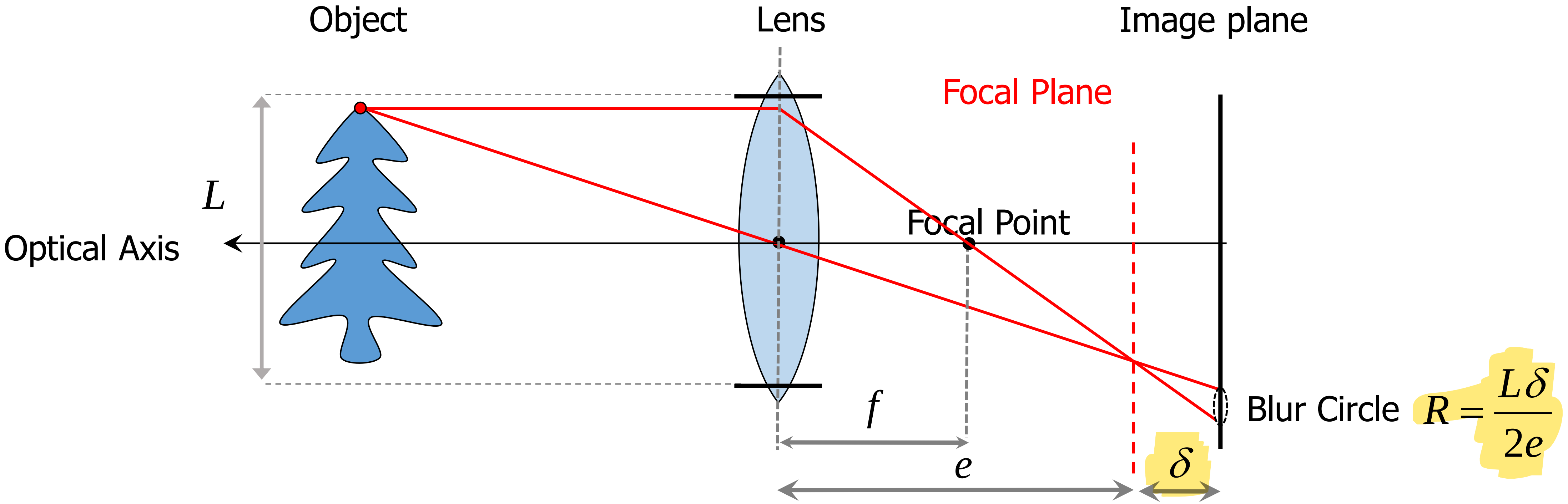
“In focus”

- For a given point on the object, there is a specific distance between the lens and the film, at which the object appears **in focus** in the image
- Other points project to a **blur circle** in the image



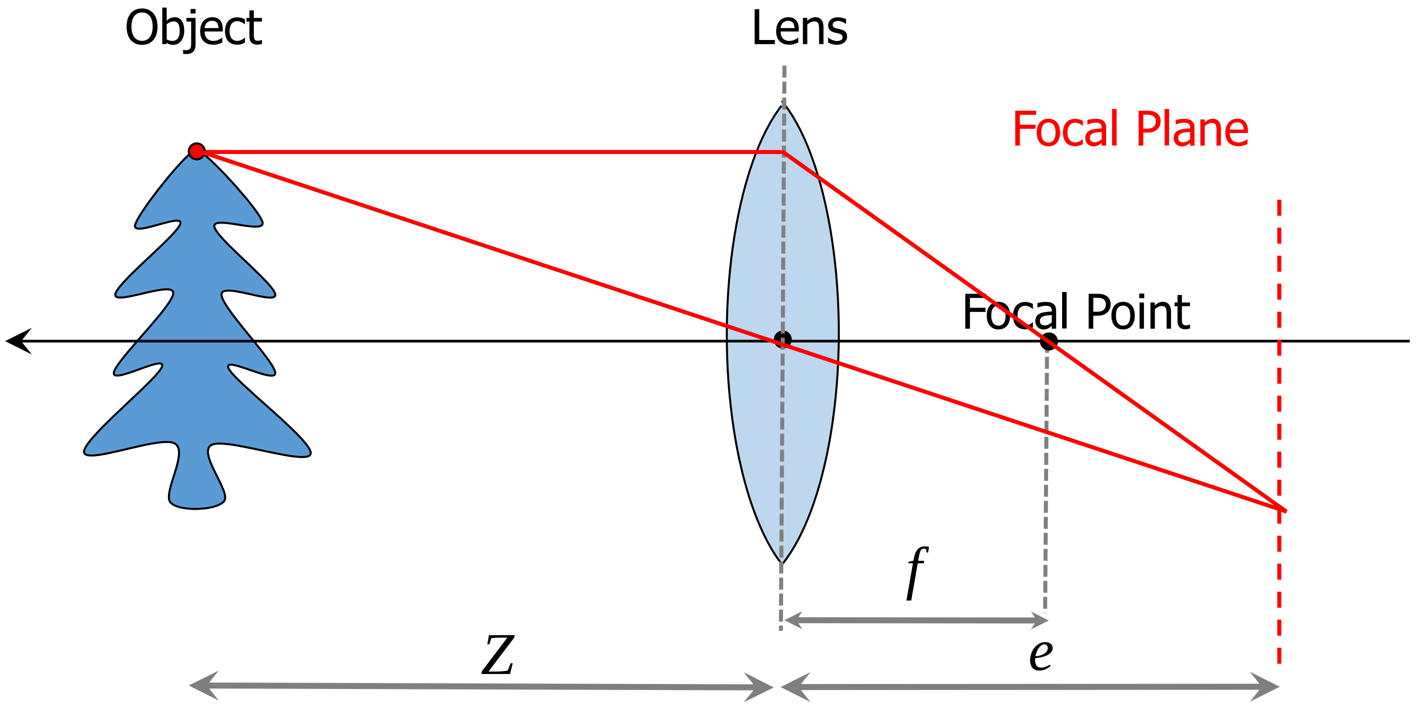
Blur Circle

- The blur circle has radius: $\boldsymbol{R = L\delta/(2e)}$
  - Observe how both **small aperture ($\boldsymbol{L}$) and $\boldsymbol{\delta}$** yield a **small blur circle ($\boldsymbol{R}$)**
  - To capture a sharp image, we must **adjust the camera settings** such that $\boldsymbol{R}$ remains **smaller** than the **1 pixel**



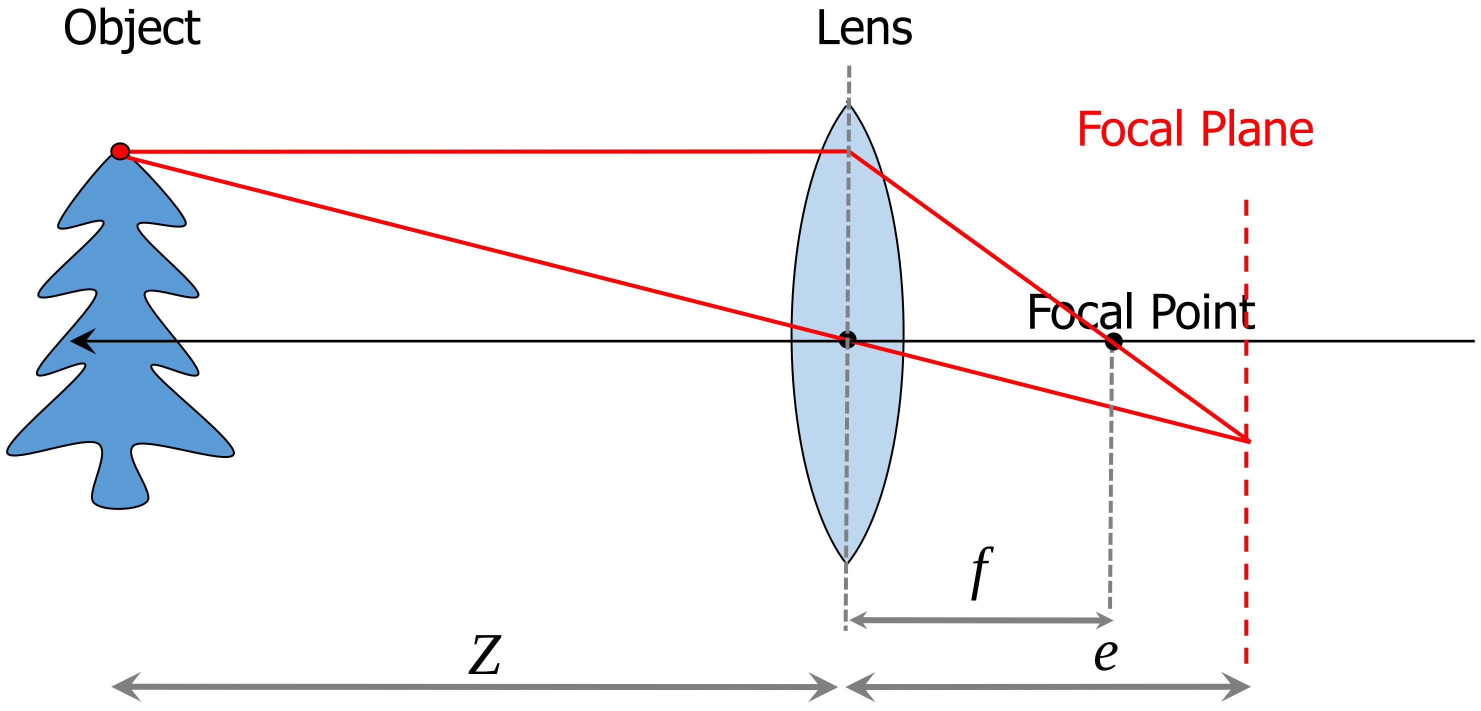
The Pin-hole approximation

- What happens if $z \gg f$ and $z \gg L$?



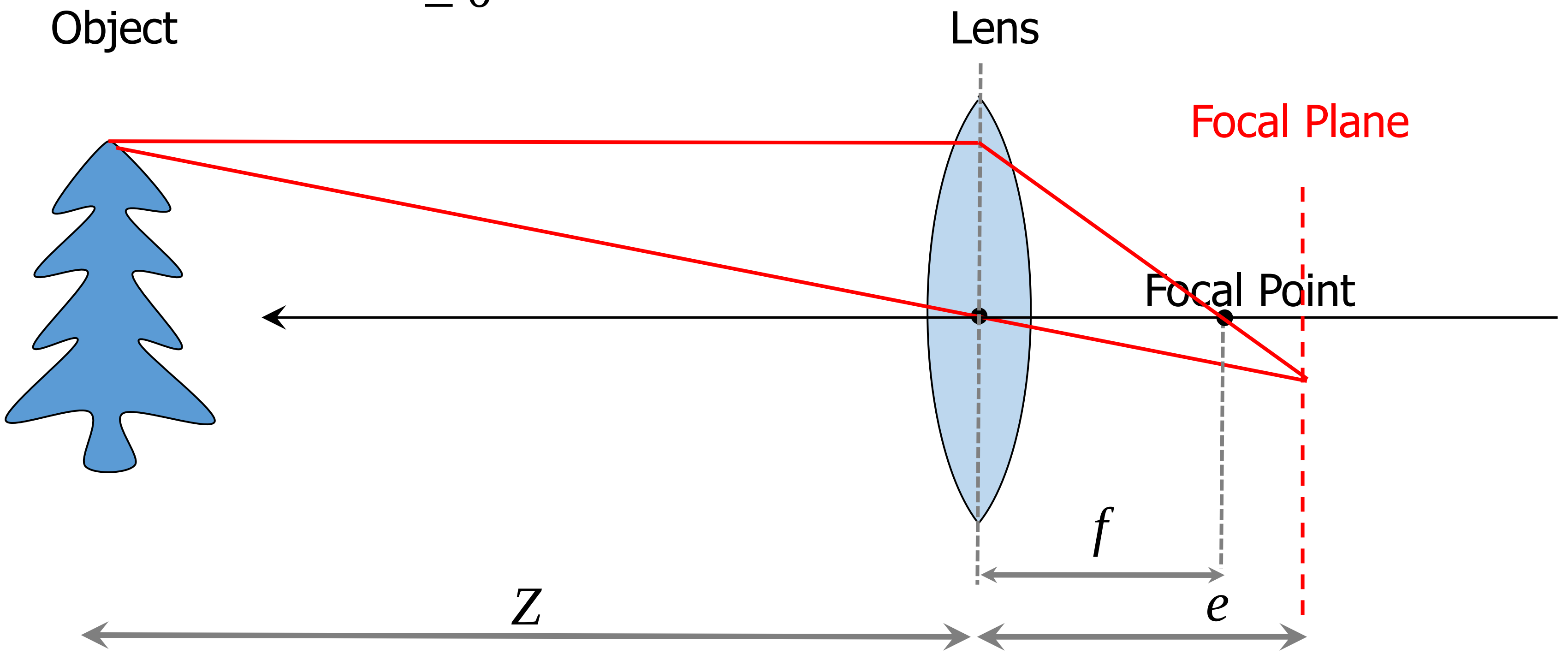
The Pin-hole approximation

- What happens if $z \gg f$ and $z \gg L$?



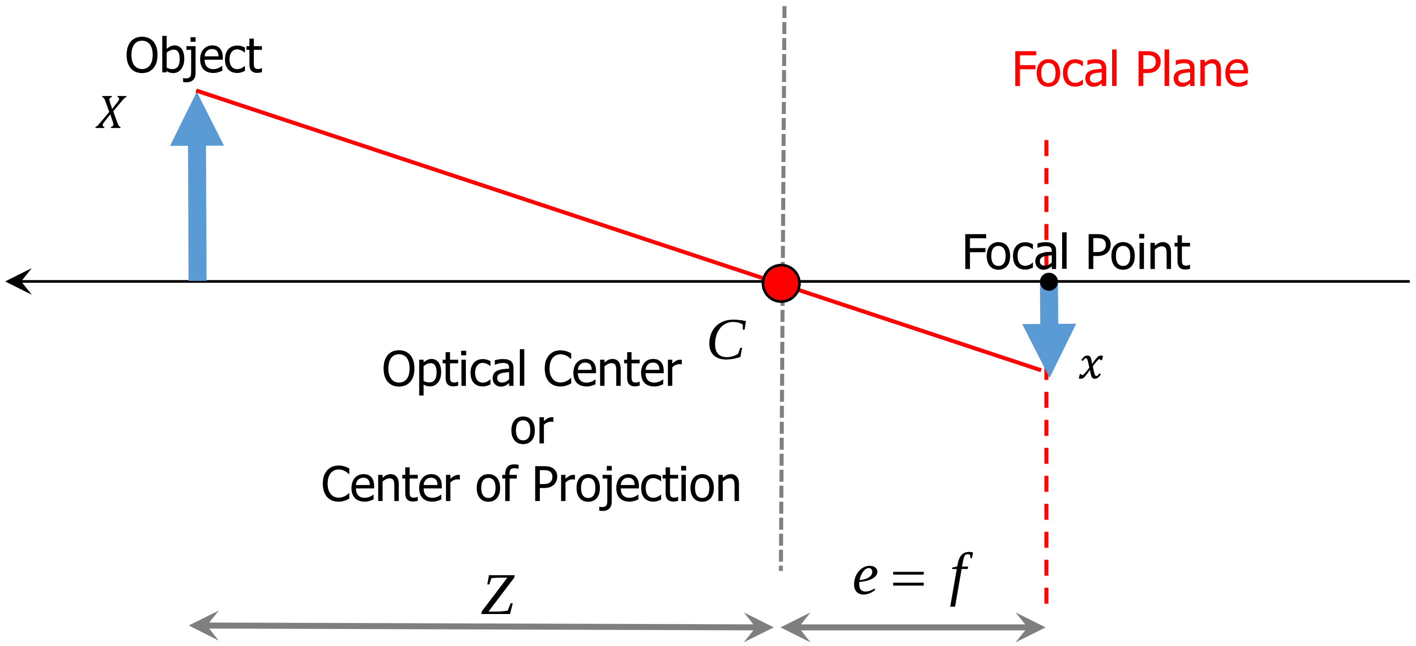
The Pin-hole approximation

- What happens if $z \gg f$ and $z \gg L$?
- We observe that $\frac{1}{f} = \underbrace{\boxed{\frac{1}{z}}}_{\cong 0} + \frac{1}{e} \Rightarrow \frac{1}{f} \approx \frac{1}{e} \Rightarrow \boxed{f \approx e} \Rightarrow$ focal plane approaches the focal point



The Pin-hole approximation

- This is known as **Pinhole Approximation**
- The relation between the image and object becomes: $-\frac{x}{X} = \frac{f}{Z} \Rightarrow \boxed{x = -f\frac{X}{Z}}$
- This is called **Perspective Projection**



Perspective effects

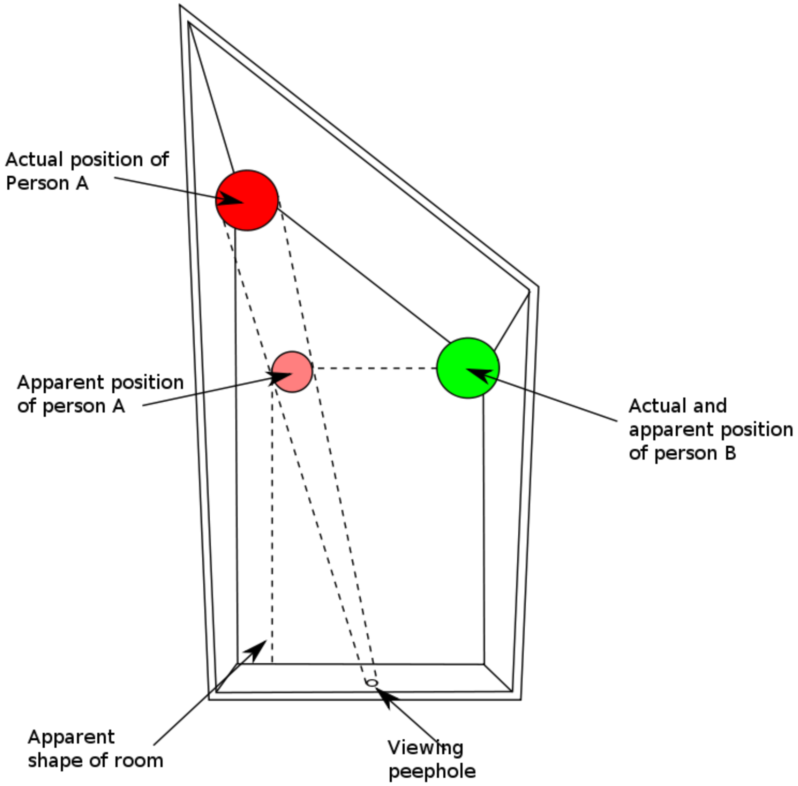
Far away objects appear smaller, with **size inversely proportional to distance**

Perspective can be used as a prior

Perspective gives us very strong depth cues and, thanks to experience (e.g., Manhattan world assumption), we **can perceive a 3D scene by viewing its 2D** representation (i.e. the image)

Playing with Perspective: the Ames room

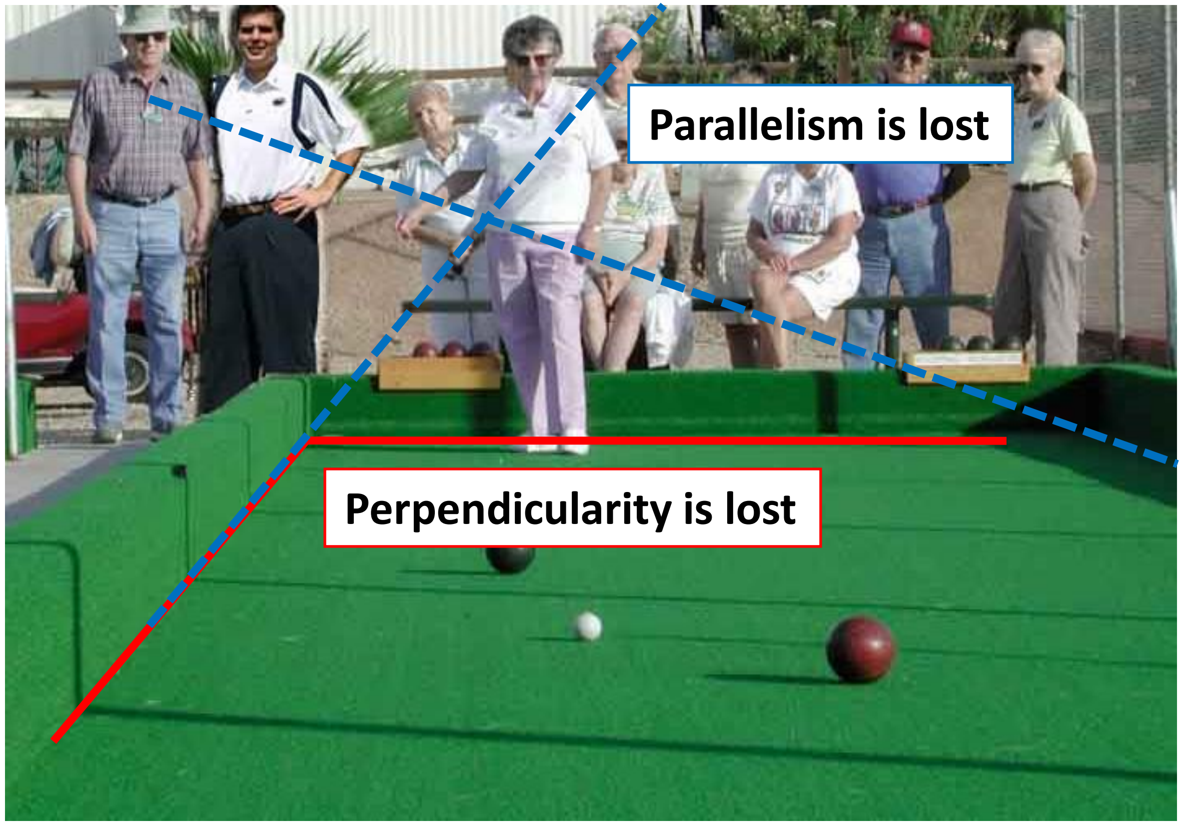
- However, our perception of a 3D scene can be fooled. An example is the Ames room (btw, check out the **Ames room** in the **Technorama** science museum in Winterthur)



Ames room documentary by neuroscientist Dr. V.S. Ramachandran: https://youtu.be/Ttd0YjXF0no

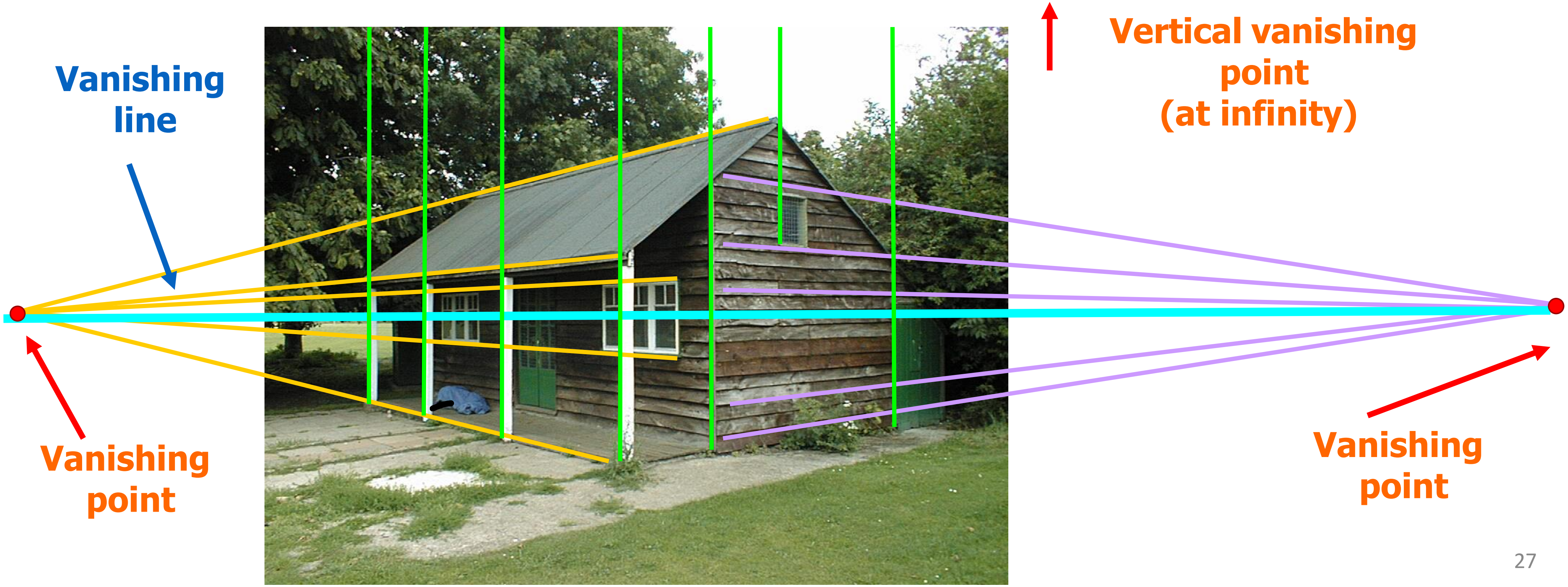
Perspective Projection: what is preserved or lost?

- **Straight lines** are still **straight**
- **Lengths and angles** are **not preserved**



Vanishing points and lines

- **Parallel lines** intersect at a "**vanishing point**" in the image
- **Parallel planes** intersect at a "**vanishing line**" in the image
- Notice that vanishing points can fall both inside or outside the image

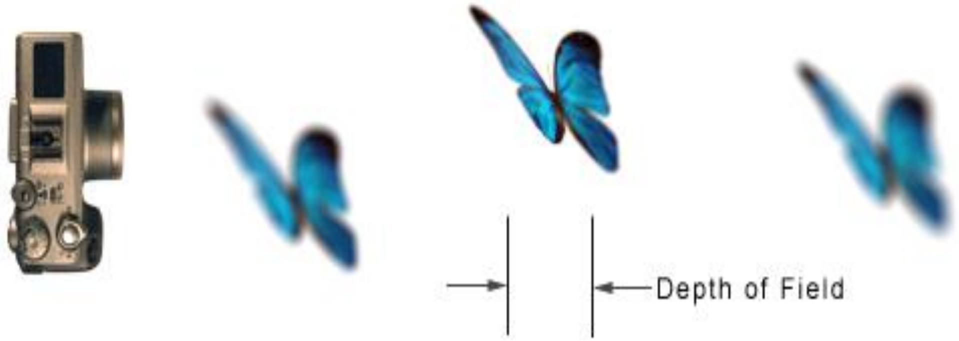


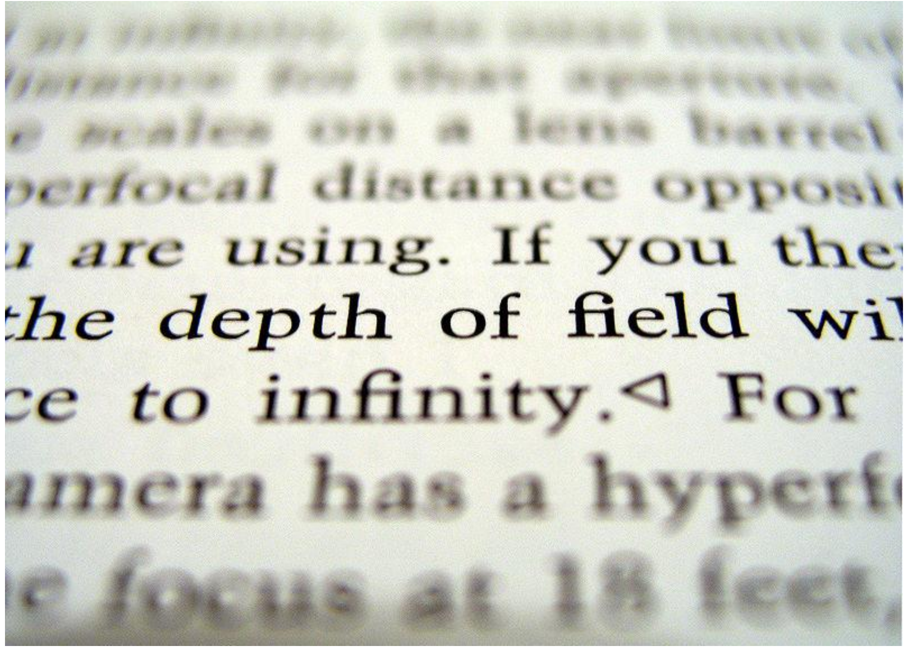
Today's Outline

- Image Formation
- Other camera parameters
- Digital camera
- Perspective camera model
- Lens distortion

Focus and Depth of Field

- **Depth of Field** is the distance between the **nearest and farthest objects** in a scene that appear **acceptably sharp** in an image

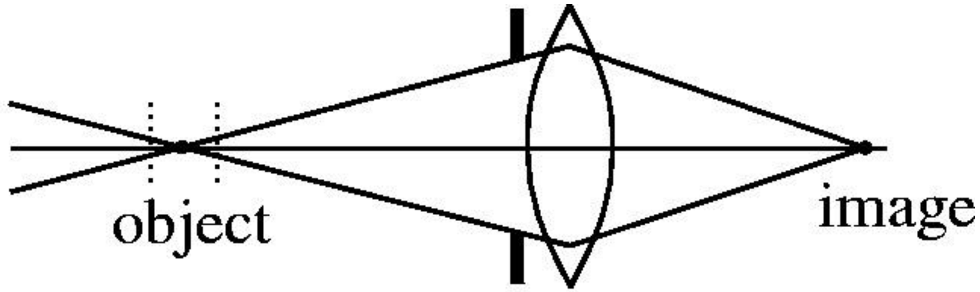


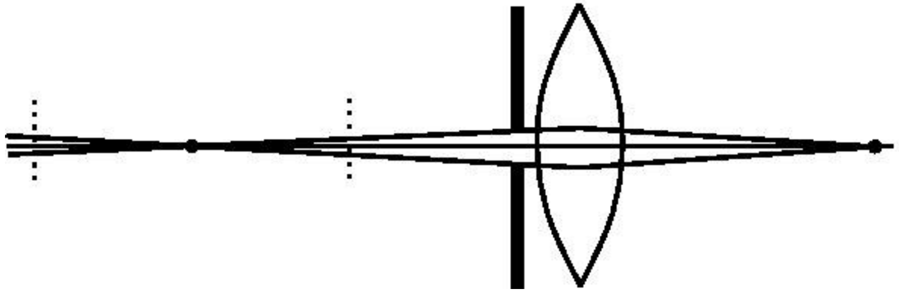


Depth of field

Focus and Depth of Field

- A **smaller aperture increases the depth of field** but **reduces** the **amount of light** into the camera: recall the definition of blur circle (it reduces with aperture)

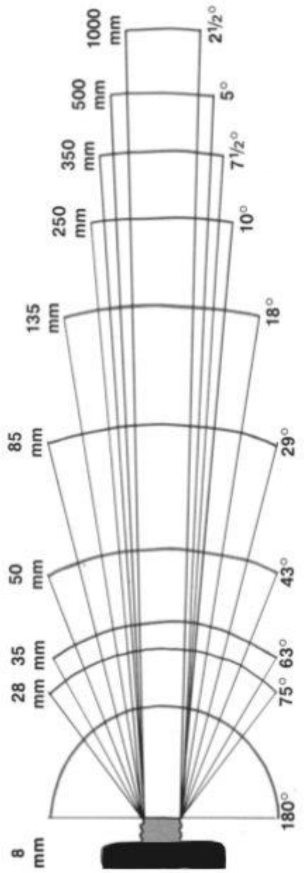




What is the depth of field of an ideal pinhole camera?

Field of View (FOV)

- The FOV is the angular portion of 3D scene seen by the camera

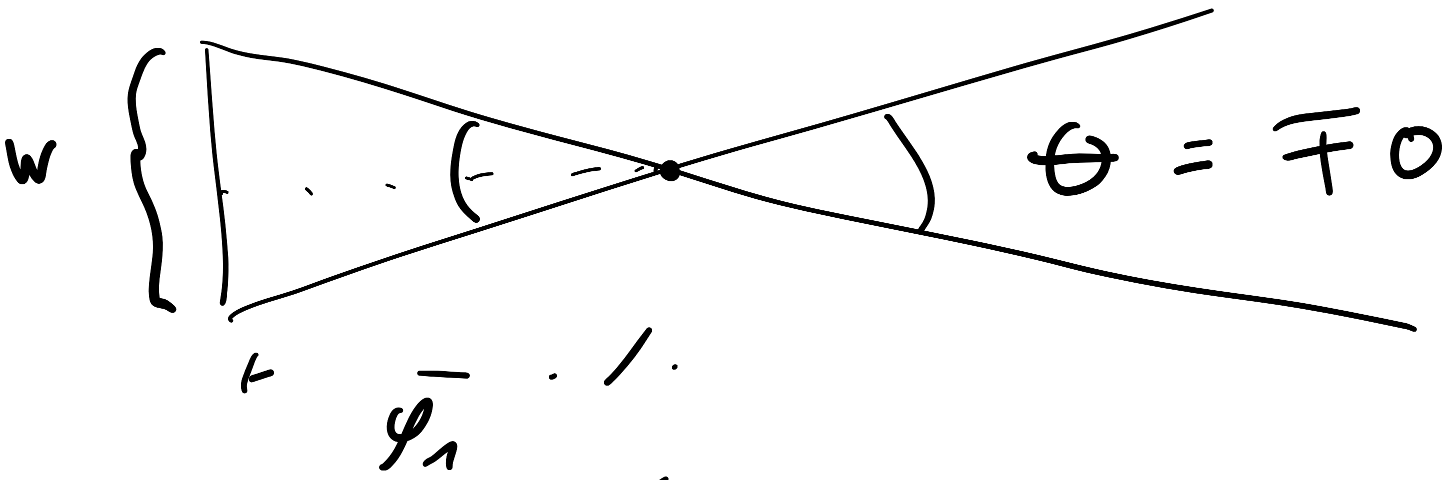


28 mm lens, 65.5° × 46.4°

50 mm lens, 39.6° × 27.0°

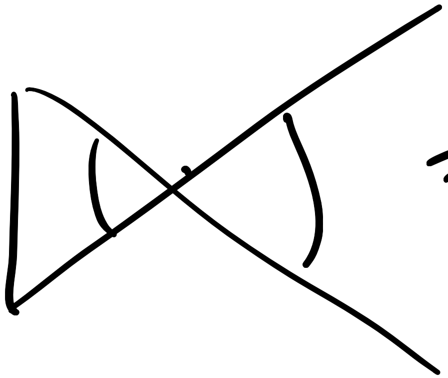
70 mm lens, 28.9° × 19.5°

210 mm lens, 9.8° × 6.5°

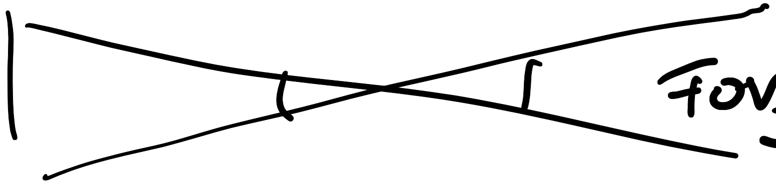


$\theta = FOV_1$

$$\boxed{\operatorname{tg}\frac{\theta}{2} = \frac{W}{2\rho}}$$



$FOV_2 > FOV_1$

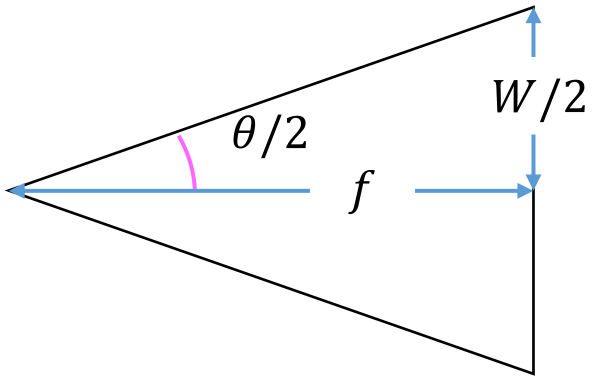


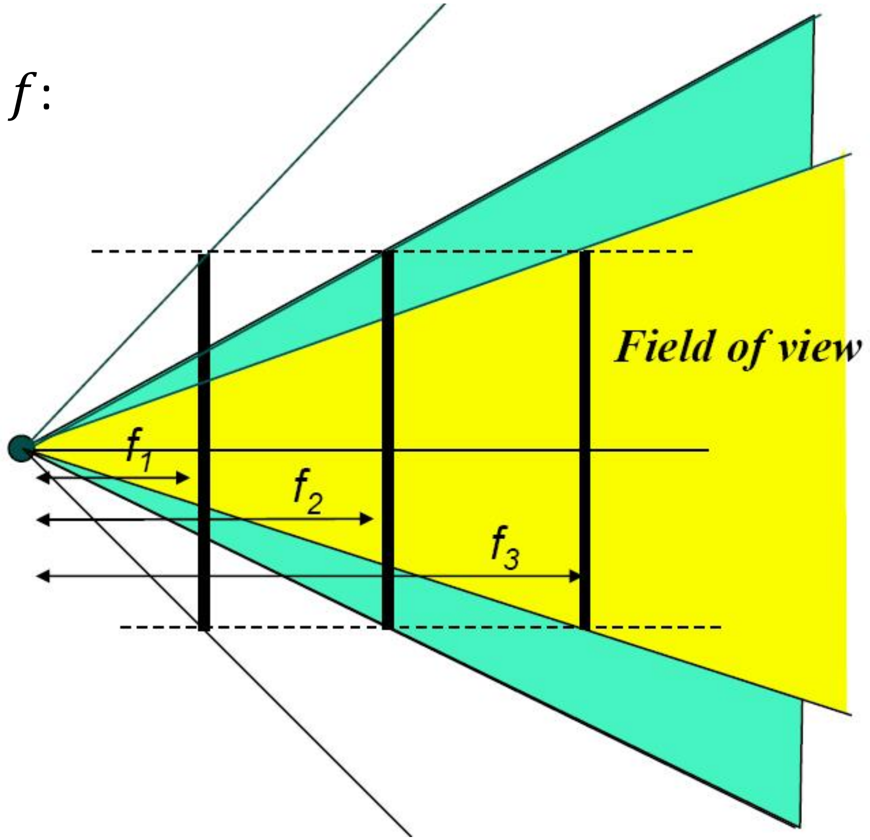
FOV_3

Field of View (FOV)

- As focal length f gets smaller, image becomes more *wide angle*
- As focal length f gets larger, image becomes more *narrow angle*
- Relation between field of view, θ, image size, W, and focal length, f:

$$\tan\frac{\theta}{2} = \frac{W}{2f} \rightarrow f = \frac{W}{2}\left[\tan\frac{\theta}{2}\right]^{-1}$$



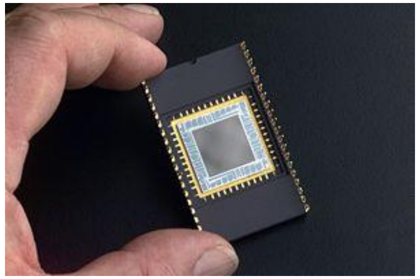


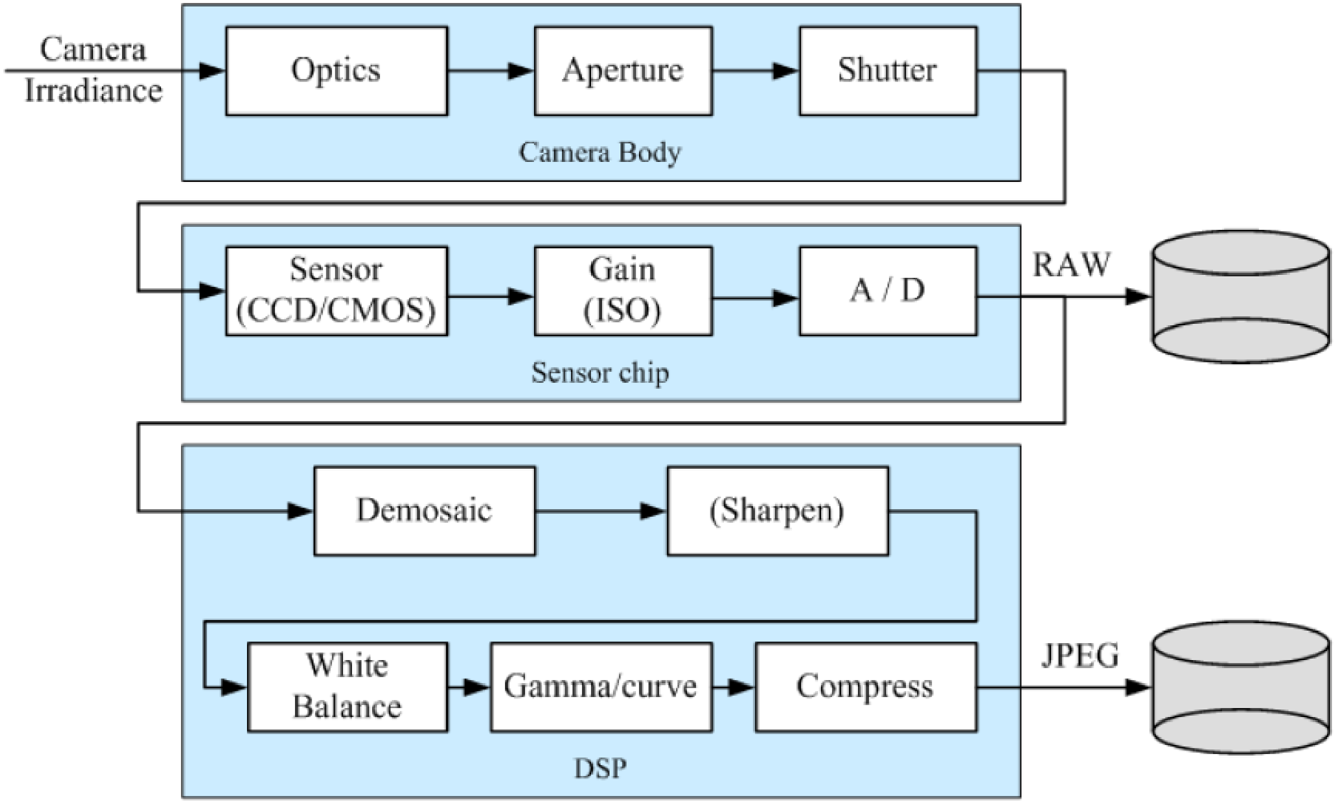
Today's Outline

- Image Formation
- Other camera parameters
- Digital camera
- Perspective camera model
- Lens distortion

Digital camera

- In a digital camera the **film** is an array of photodiodes (CCD or CMOS) that convert photons (light energy) into electrons





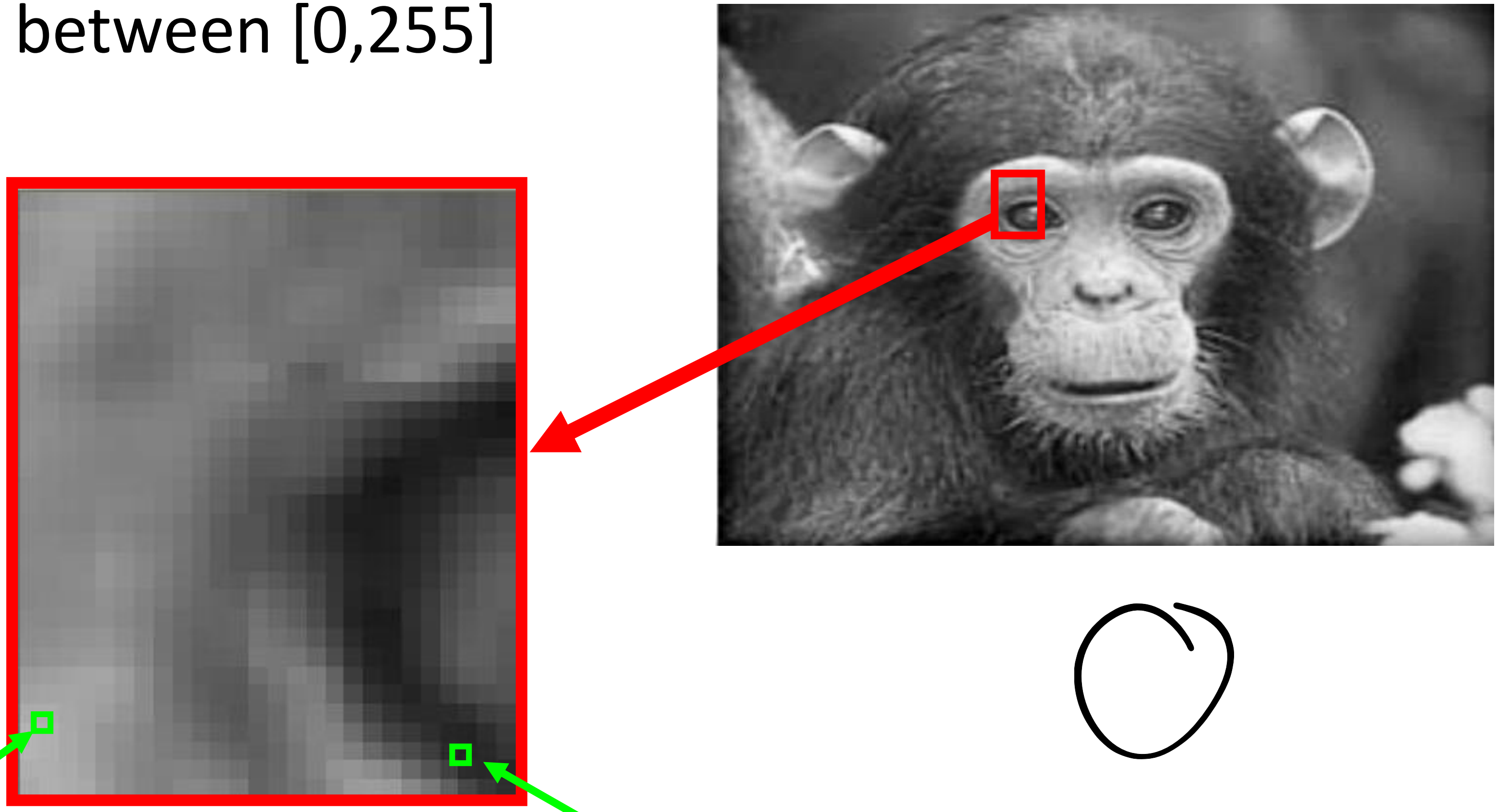
Digital images

Pixel Intensity with 8 bits ranges between [0,255]

j=1

width 640

i=1



height 480

im[176][201] has value 164

im[194][203] has value 37

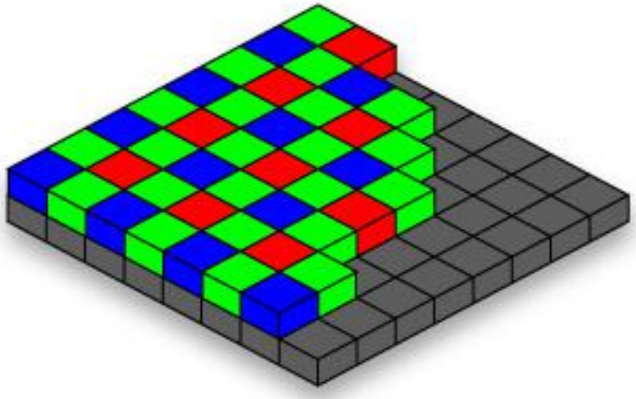
NB. Matlab coordinates: [rows, cols]; C/C++ [cols, rows]

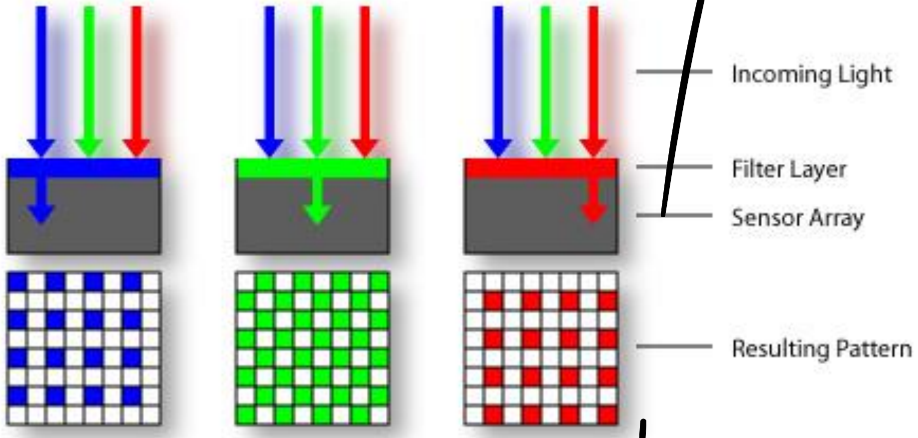
Color sensing in digital cameras

- The **Bayer pattern** (invented by Bayer in 1976, who worked at Kodak) places interleaved RGB filters over the pixel array
- The reason why the number of green filters is twice as that of red and blue ones is because the **luminance** signal is mostly **determined by green** values and the **human visual system** is much more **sensitive** to **spatial differences in luminance** than in chrominance.

For each pixel, the missing color components can be estimated from neighboring values by interpolation (**demosaicing**)

Bayer grid

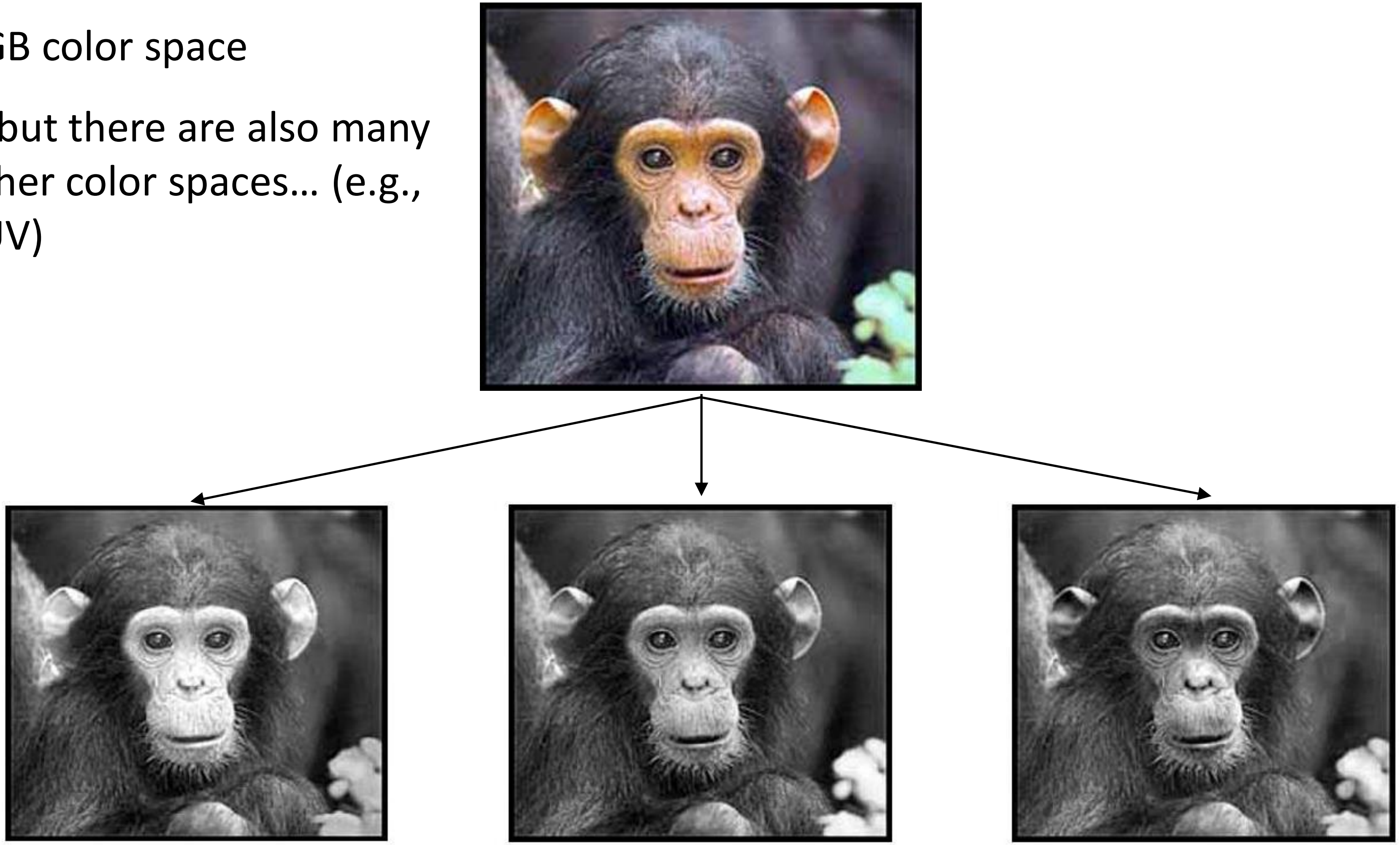




Color sensing in digital cameras

RGB color space

… but there are also many other color spaces… (e.g., YUV)



R G B

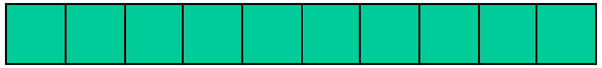
Rolling vs Global Shutter Camera

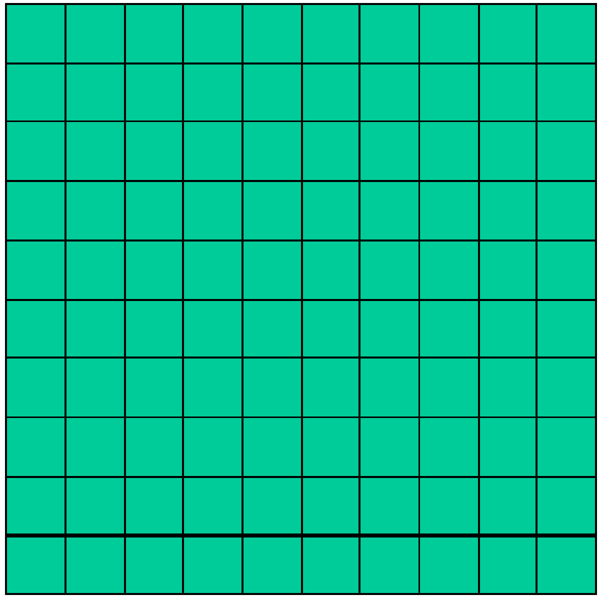
Rolling Shutter

- **Rows** of pixels are **exposed and read at different times,** one after the other
- May **distort** (skew) **moving objects**
- **Cheap**

Global Shutter

- **All pixels** are **exposed simultaneously**
- **No distortion** of moving objects
- More **costly**





Rolling vs Global Shutter Camera

- Rolling Shutter cameras may distort (skew) moving objects
- Global Shutter cameras do not have this problem

Rolling shutter

Global shutter

Cool application of rolling shutter cameras

- High speed wave-form capturing from a guitar

Recorded from an iPhone 4 (rolling shutter):
https://youtu.be/TKF6nFzpHBU

An example camera datasheet

mvBlueFOX-IGC / -MLC

Technical Details

Sensors

mvBlueFOX-IGC mvBlueFOX-MLC		Resolution (H x V pixels)	Sensor size (optical)	Pixel size (µm)	Frame rate	Sensor technology	Readout type	ADC resolution / output in bits	Sensor
-200w [1,2]	G/C	752 x 480	1/3"	6 x 6	90	CMOS	Global	10 → 10 / 8	Aptina MT9V
-202b	G/C	1280 x 960	1/3"	3.75 x 3.75	24.6	CMOS	Global	10 → 10 / 8	Aptina MT9M
-202d [1]	G/C	1280 x 960	1/3"	3.75 x 3.75	24.6	CMOS	Rolling	10 → 10 / 8	Aptina MT9M
-205 [2]	G/C	2592 x 1944	1/2.5"	2.2 x 2.2	5.8	CMOS	Global Reset	10 → 10 / 8	Aptina MT9P

[1] High Dynamic Range (HDR) mode supported

[2] Software trigger supported

Sample: **mvBlueFOX-IGC200wG** means version with housing and 752 x 480 CMOS gray scale sensor.
mvBlueFOX-MLC200wG means single-board version without housing and with 752 x 480 CMOS gray scale sensor.

Nano-drone of my lab equipped with this camera

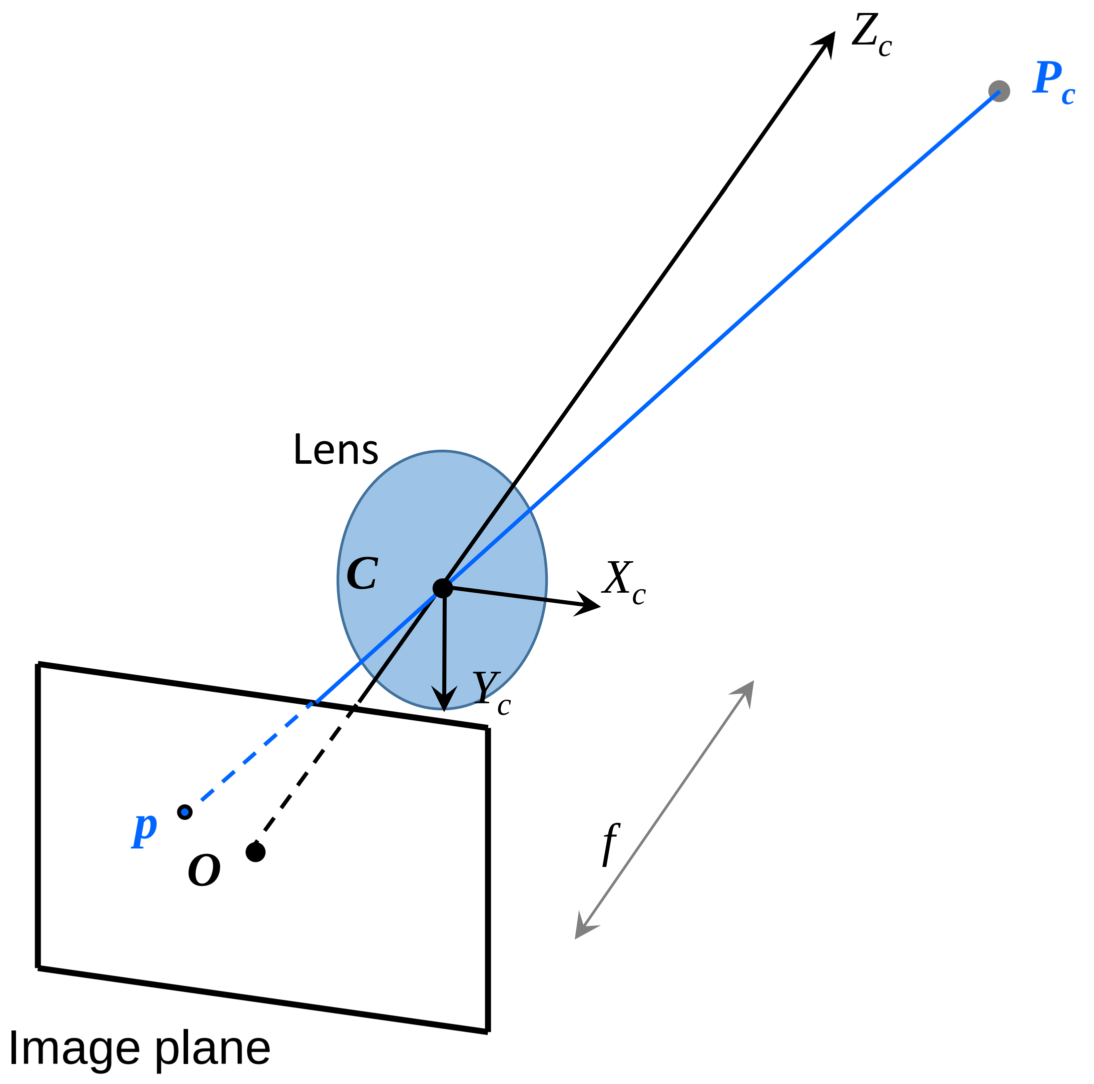
Today’s Outline

- Image Formation
- Other camera parameters
- Digital camera
- Perspective camera model
- Lens distortion

Perspective Camera

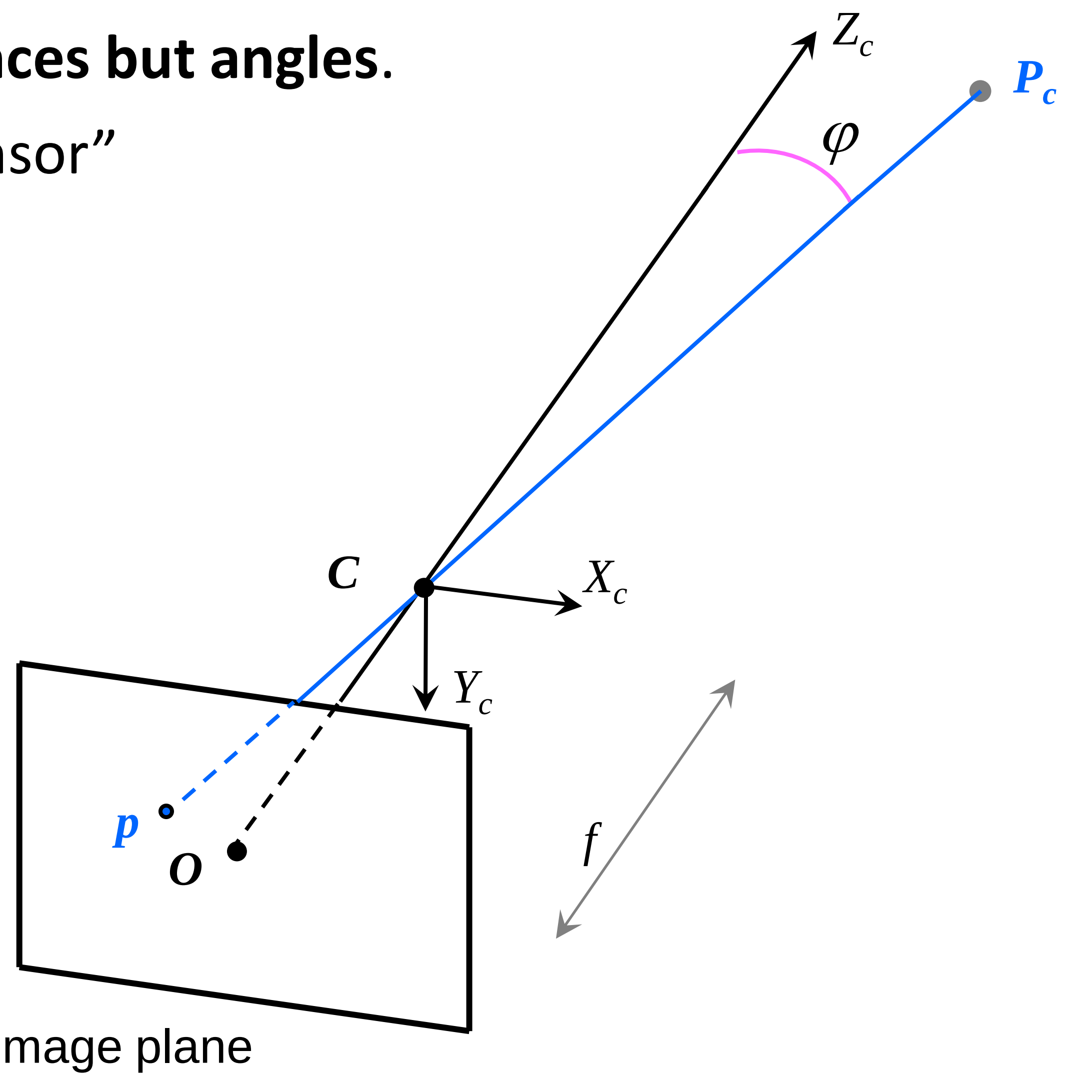
Nomenclature:

- ***C*** = optical center
  = center of the lens
  = center of projection
- X_c, Y_c, Z_c = axes of the camera reference frame
- Z_c = optical axis
- ***O*** = principal point
  = intersection of optical axis and image plane



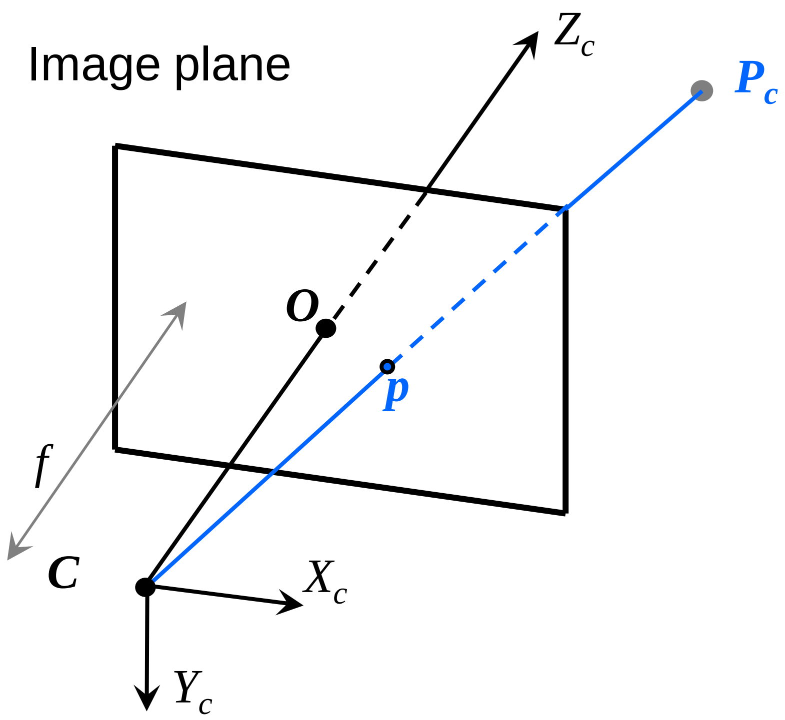
Perspective Camera

- Note that **a camera does not measure distances but angles**.
- So, we call it a “bearing sensor” or “angle sensor”



Perspective Camera

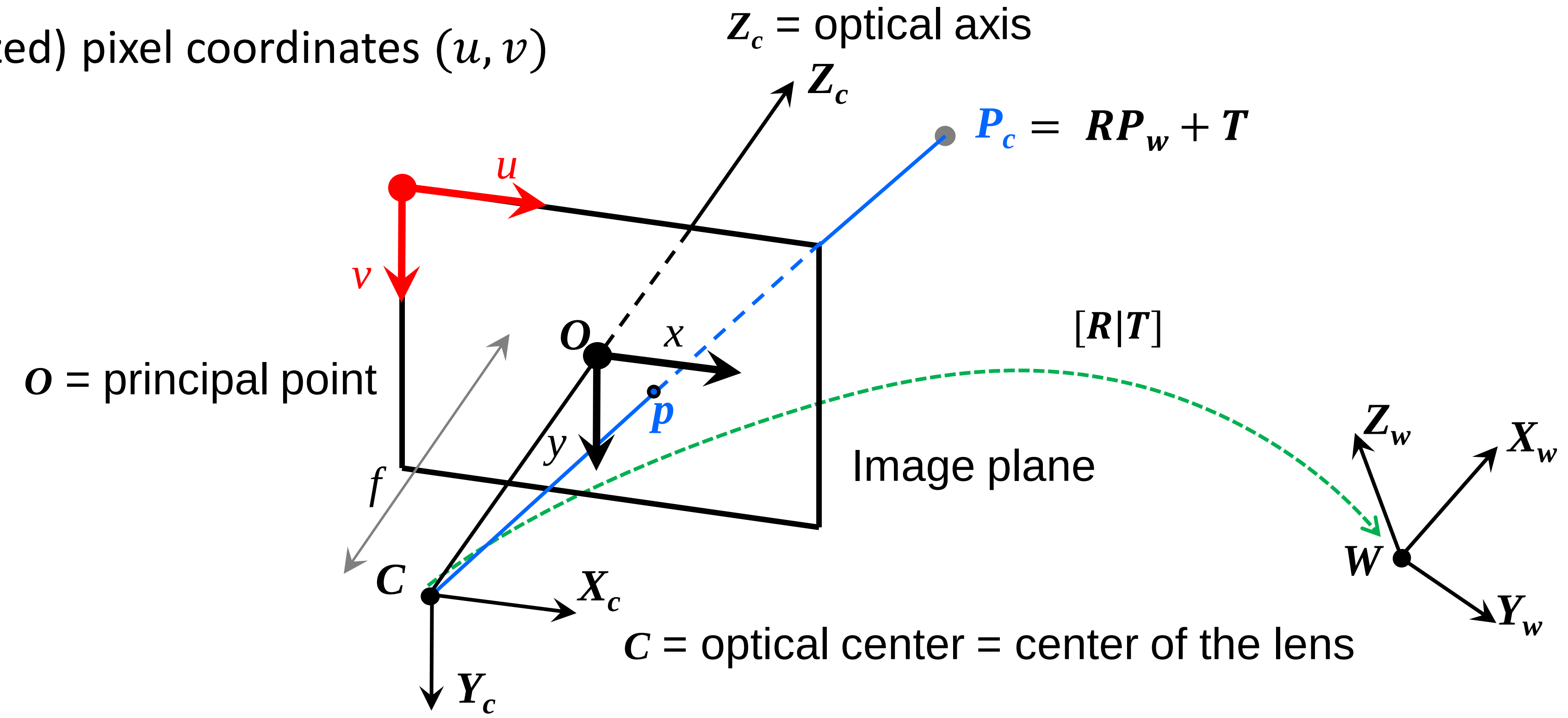
- For convenience, the **image plane** is usually **represented in front** of the lens, $\boldsymbol{C}$, such that the image preserves the same orientation (i.e. not flipped)



From World to Pixel coordinates

Goal: Find pixel coordinates $(\boldsymbol{u}, \boldsymbol{v})$ of the projection of world point $\boldsymbol{P_W}$ onto the image plane:

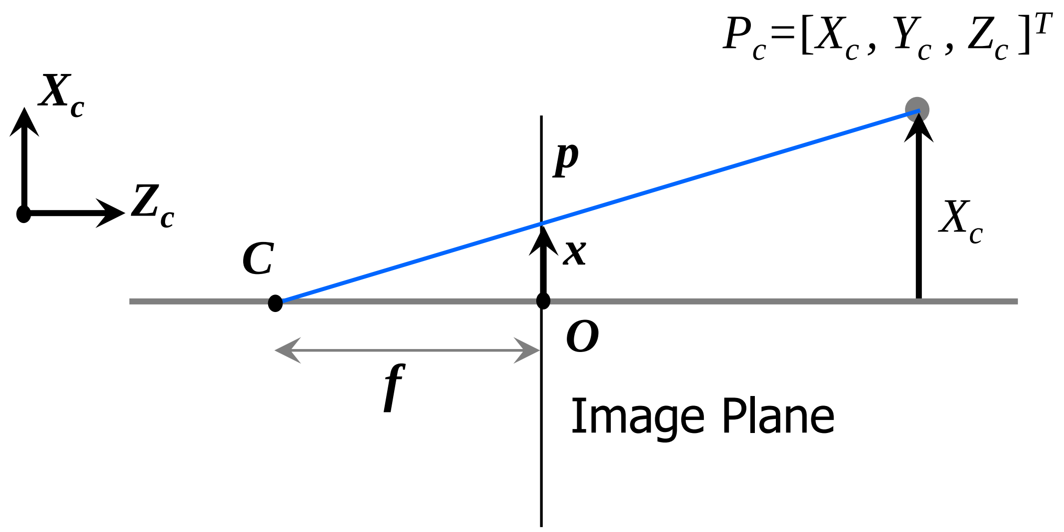
- Convert world point P_W to camera point P_C through rigid body transformation $[R, T]$
- Convert P_C to image-plane coordinates (x, y)
- Convert (x, y) to (discretized) pixel coordinates (u, v)



Perspective Projection – 1/4

From Camera frame to image plane coordinates $(\boldsymbol{x}, \boldsymbol{y})$

- The Camera point $P_c=[X_c, Y_c, Z_c]^T$ projects to $p=(x, y)$ onto the image plane
- From similar triangles: $\frac{x}{f}=\frac{X_c}{Z_c} \Rightarrow x=\frac{fX_c}{Z_c}$
- Similarly, for y: $\frac{y}{f}=\frac{Y_c}{Z_c} \Rightarrow y=\frac{fY_c}{Z_c}$



Perspective Projection – 2/4

From image plane coordinates (x, y) to pixel coordinates (u, v)

- Let $O = (u_0, v_0)$ be the pixel coordinates of the camera optical center
- Let k_u, k_v be the pixel conversion factors (inverse of the pixel-size along x and y)

$$u = u_0 + k_u x \quad \rightarrow \quad u = u_0 + \frac{k_u f X_C}{Z_C}$$

$$v = v_0 + k_v y \quad \rightarrow \quad v = v_0 + \frac{k_v f Y_C}{Z_C}$$

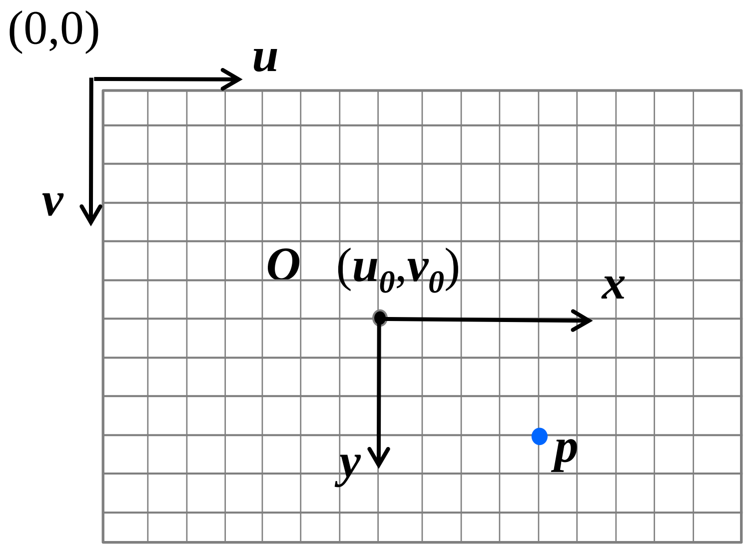


Image plane

Perspective Projection – 2/4

From image plane coordinates (x, y) to pixel coordinates (u, v)

- Let $O = (u_0, v_0)$ be the pixel coordinates of the camera optical center
- Let k_u, k_v be the pixel conversion factors (inverse of the pixel-size along x and y)

$$u = u_0 + k_u x \rightarrow u = u_0 + \frac{\alpha_u X_C}{Z_C}$$

$$v = v_0 + k_v y \rightarrow v = v_0 + \frac{\alpha_v Y_C}{Z_C}$$

Focal lengths
(expressed in **pixels**)

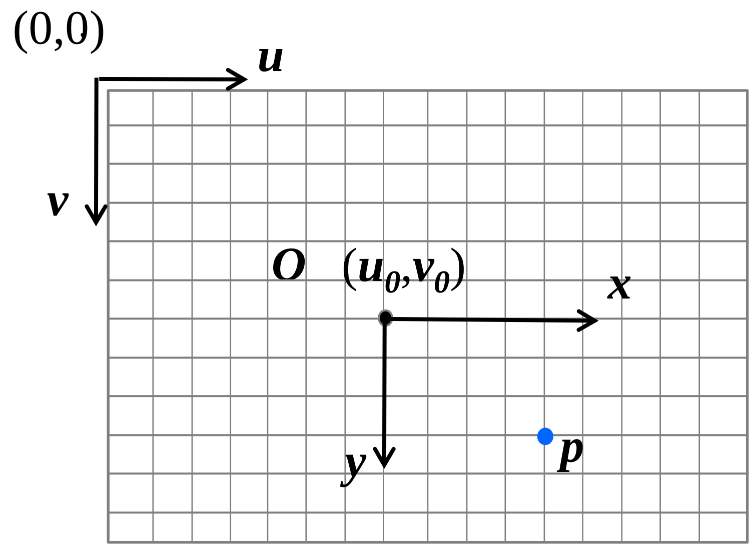


Image plane

$$u = u_0 + \alpha_u \frac{x_c}{z_c} \Longrightarrow z_c u = z_c u_0 + \alpha_u x_c$$

$$v = v_0 + \alpha_v \frac{y_c}{z_c} \Longrightarrow z_c v = z_c v_0 + \alpha_u y_c$$

$$\lambda = z_c$$

$$\left.\begin{aligned} \lambda u &= \alpha_u x_c + u_0 z \\ \lambda v &= \alpha_u + y_c + v_0 z_c \\ \lambda &= z_c \end{aligned}\right\} = \underbrace{\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_u & v_0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}}$$

Perspective Projection – 3/4

- Use **Homogeneous Coordinates** for linear mapping from 3D to 2D:

$$p = \begin{pmatrix} u \\ v \end{pmatrix} \Rightarrow \tilde{p} = \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

- Using matrix form and homogeneous coordinates, the perspective projection becomes:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \boxed{\begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$

This matrix is called "**Calibration matrix**" or "**Intrinsic Parameter Matrix**" and is often denoted as $\boldsymbol{K}$

In the past it was common to assume a skew factor ($K_{12} \neq 0$) to account for possible skew in the pixel manufacturing process. However, the camera manufacturing process today is so good that we can safely assume $K_{12} = 0$ and $\alpha_u = \alpha_v$ (i.e., square pixels).

Perspective Projection – 3/4

- Use **Homogeneous Coordinates** for linear mapping from 3D to 2D:

$$p = \begin{pmatrix} u \\ v \end{pmatrix} \qquad \tilde{p} = \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

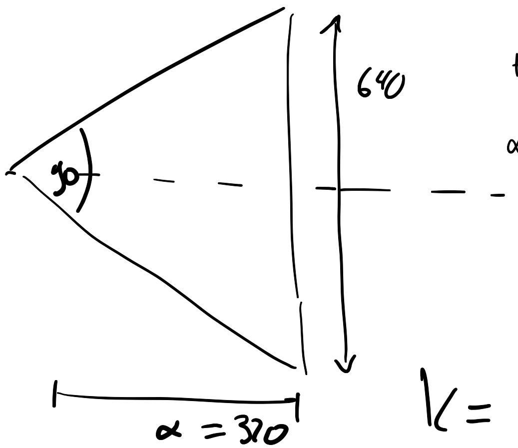
- Using matrix form and homogeneous coordinates, the perspective projection becomes:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$

Compact form

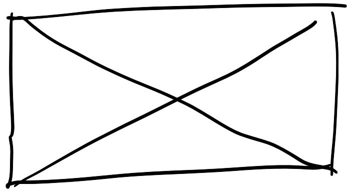
Exercise 1

- **Determine the Intrinsic Parameter Matrix** ($\boldsymbol{K}$) of a digital camera with an image size 640×480 pixels and a horizontal field of view of $90°$
- Assume square pixels and the principal point as the center of the diagonals
- What is the vertical field of view?
- What's the projection on the image plane of $P_c = [1, 1, 2]^{\mathsf{T}}$



$tg\left(\frac{\sigma}{2}\right) = \frac{u}{2\alpha}$

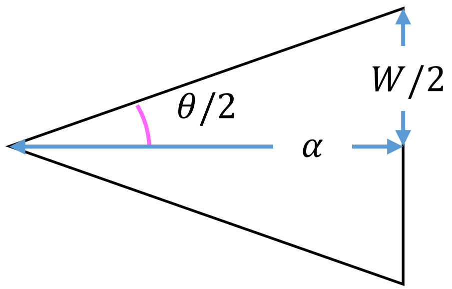
$\alpha_u = \alpha_v = 320$ pixels



$$K = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 320 & 0 & 340 \\ 0 & 320 & 240 \\ 0 & 0 & 1 \end{bmatrix}.$$

Exercise 1 - Solution

- Recall:



$$\tan\frac{\theta}{2} = \frac{W}{2\alpha} \quad \rightarrow \quad \alpha = \frac{W}{2}\left[\tan\frac{\theta}{2}\right]^{-1} = \frac{640}{2\tan\frac{\pi/2}{2}} = 320\ [pixels]$$

$\rightarrow$ α_u = α_v $= \alpha = 320\ pixels$ (because pixels are square)

- Because the Principal Point is center of the diagonals $\rightarrow O = (u_0, v_0) = \left(\frac{640}{2}, \frac{480}{2}\right) = (320\,, 240)$

$$\rightarrow \quad K = \begin{bmatrix} 320 & 0 & 320 \\ 0 & 320 & 240 \\ 0 & 0 & 1 \end{bmatrix}$$

- Vertical FOV: $\theta_V = 2\tan^{-1}\frac{H}{2\alpha} = 2\tan^{-1}\frac{480}{2\cdot 320} = 73.74°$

Exercise 1 - Solution

- What's the projection on the image plane of $P_c = [1, 1, 2]^{\mathrm{T}}$

- **Solution 1** (using **Perspective Projection Equation in plane form**)

$$u = u_0 + \frac{\alpha X_C}{Z_C} = 320 + \frac{320 \cdot 1}{2} = 480 \ [pixels]$$

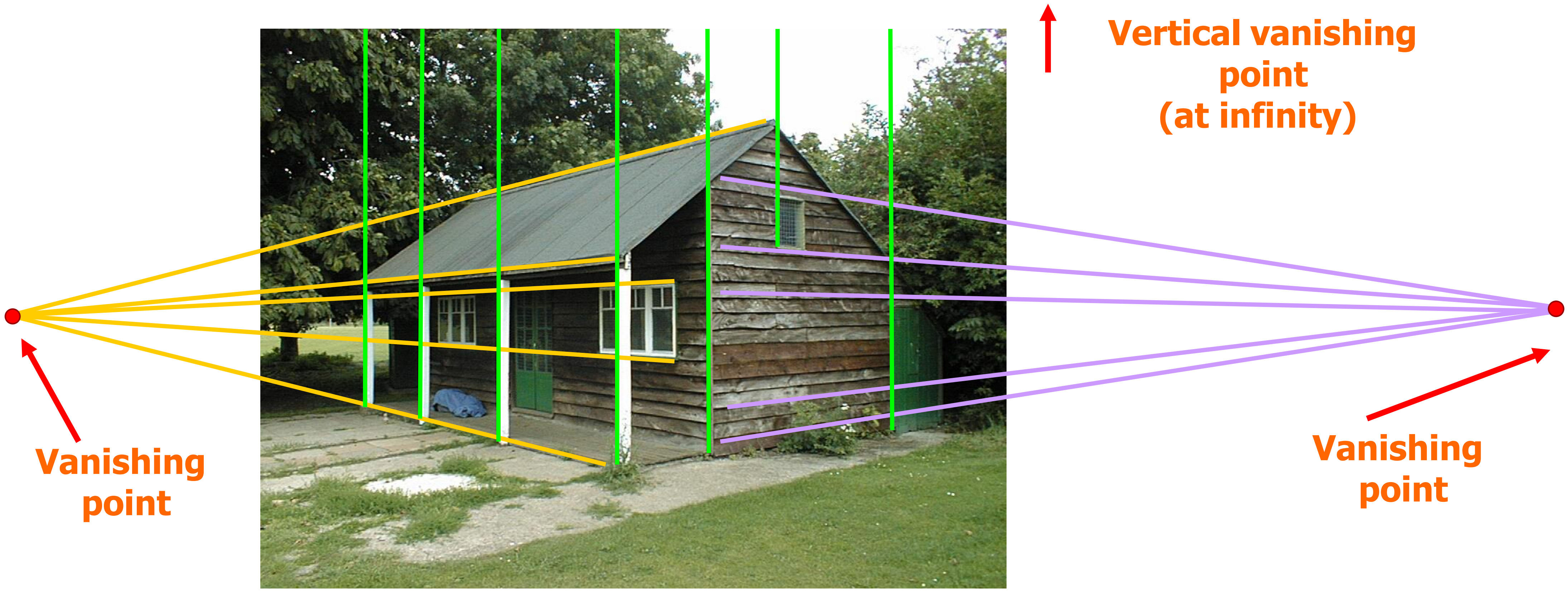
$$v = v_0 + \frac{\alpha Y_C}{Z_C} = 240 + \frac{320 \cdot 1}{2} = 400 \ [pixels]$$

- **Solution 2** (using **Perspective Projection Equation in matrix form**)

$$\begin{bmatrix} \lambda u \\ \lambda v \\ \lambda \end{bmatrix} = \begin{bmatrix} 320 & 0 & 320 \\ 0 & 320 & 240 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 960 \\ 800 \\ 2 \end{bmatrix} \rightarrow \begin{array}{l} u = \dfrac{960}{2} = 480 \ [pixels] \\ \\ v = \dfrac{800}{2} = 400 \ [pixels] \end{array}$$

Exercise 2

- Prove that world's parallel lines intersect at a vanishing point in the camera image



Exercise 2 - Solution

- World's parallel lines with direction (l, m, n), passing through (X_0, Y_0, Z_0), are described by these equations:

$$X = X_0 + sl$$
$$Y = Y_0 + sm$$
$$Z = Z_0 + sn$$

- Let's consider the perspective projection equation in plane form:

$$u = u_0 + \alpha \frac{X}{Z} \quad , \quad v = v_0 + \alpha \frac{Y}{Z}$$

- Substitute line equations into the perspective projection equation and compute limit for $s \rightarrow \infty$

$$\lim_{s \to \infty} u_0 + \alpha \frac{X_0 + sl}{Z_0 + sn} = u_0 + \alpha \frac{l}{n} \quad , \quad \lim_{s \to \infty} v_0 + \alpha \frac{Y_0 + sm}{Z_0 + sn} = v_0 + \alpha \frac{m}{n}$$

- These are the image coordinates of the vanishing point. **Notice that they only depend on the line direction** (indeed, they do not depend on (X_0, Y_0, Z_0)).

Exercise 2 - Solution

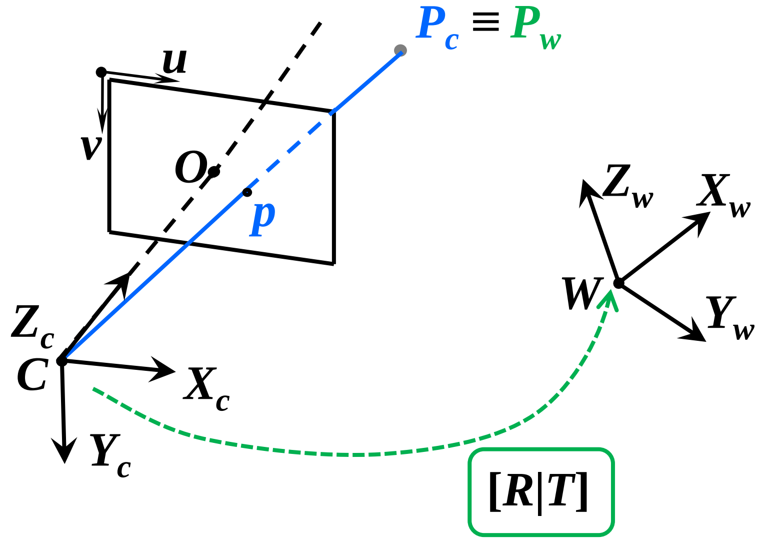
- You can verify that the solution satisfies this matrix equation. Can you show it geometrically?

$$\begin{bmatrix} \lambda u \\ \lambda v \\ \lambda \end{bmatrix} = \begin{bmatrix} \alpha & 0 & u_0 \\ 0 & \alpha & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} l \\ m \\ n \end{bmatrix}$$

Perspective Projection – 4/4

From the World frame to the Camera frame

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}$$

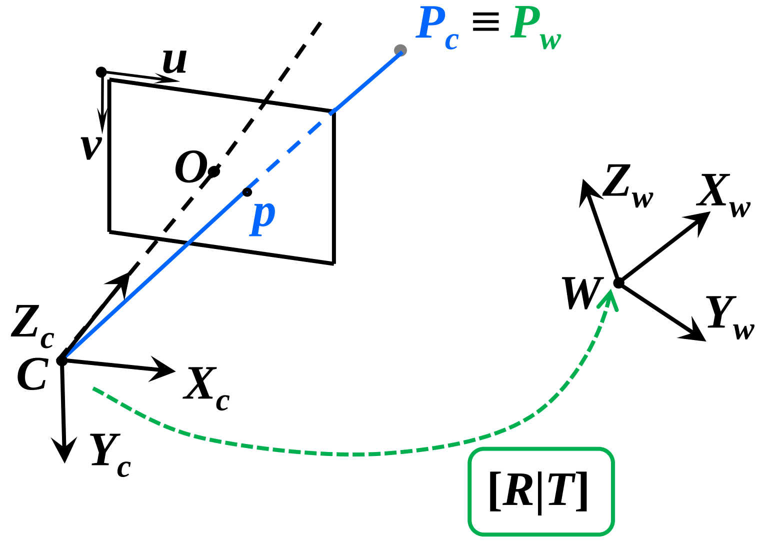


Extrinsic Parameters

Perspective Projection – 4/4

From the World frame to the Camera frame

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \left[\begin{array}{ccc|c} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{array}\right] \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \left[\begin{array}{c|c} R & T \end{array}\right] \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$



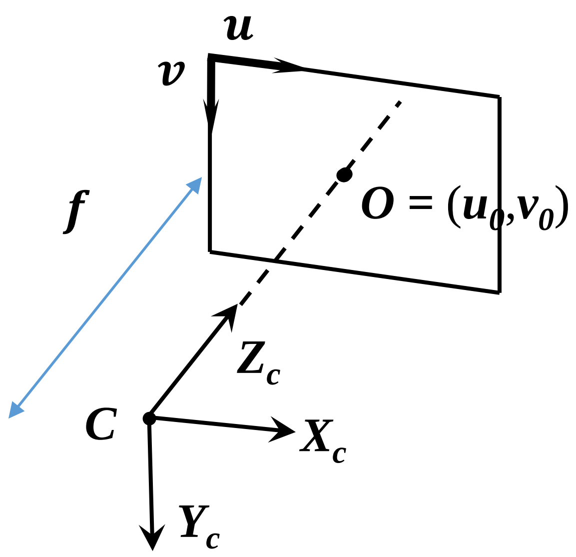
Extrinsic Parameters

By substituting the above one into the **Perspective Projection Equation**:

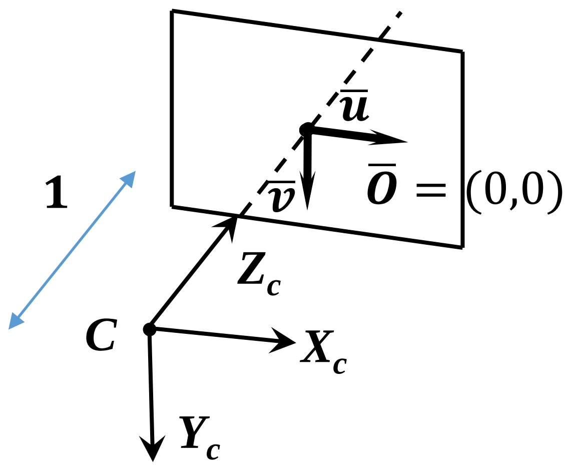
$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} \Rightarrow \boxed{\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{K\left[R \mid T\right]}_{\text{Projection Matrix (M)}} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}}$$

Perspective Projection Equation

- It is often convenient to use normalized image coordinates onto a **virtual image plane with focal length equal to 1 unit** (adimensional) **and origin of the pixel coordinates at the principal point**.



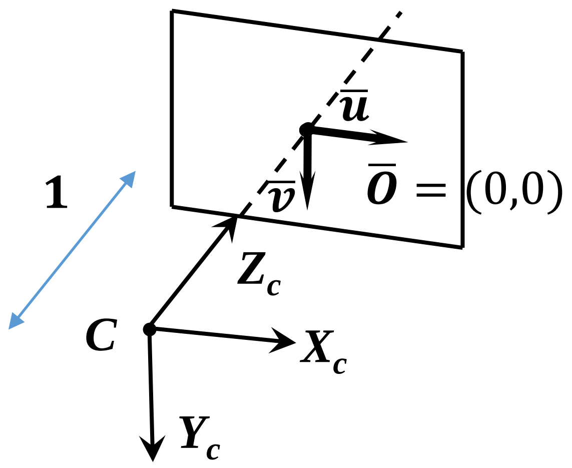
- It is often convenient to use normalized image coordinates onto a **virtual image plane with focal length equal to 1 unit** (adimensional) **and origin of the pixel coordinates at the principal point**.



Normalized image coordinates

- It is often convenient to use normalized image coordinates onto a **virtual image plane with focal length equal to 1 unit** (adimensional) **and origin of the pixel coordinates at the principal point**.
- To do this, we just pre-multiply both terms of the perspective projection equation in camera frame coordinates by K^{-1}:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} \quad \Rightarrow \lambda K^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K^{-1} \cdot K \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} \quad \Rightarrow \lambda K^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$



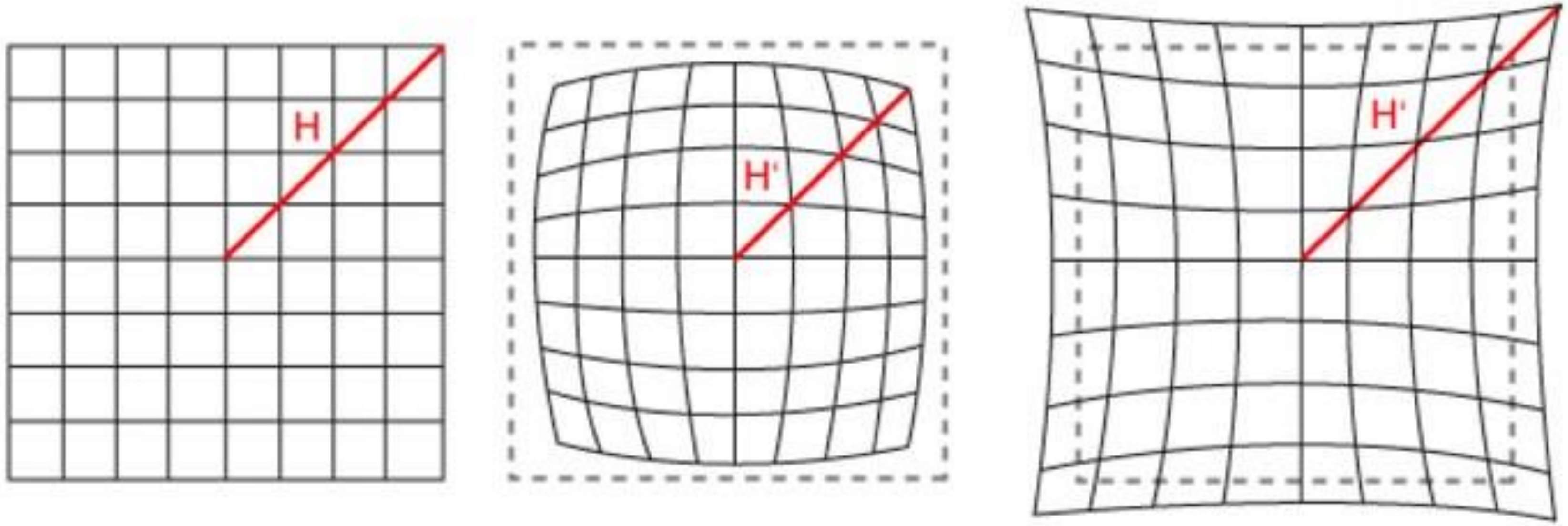
- We define the **unit-plane normalized image coordinates** $(\bar{u}, \bar{v})$:

$$\begin{bmatrix} \bar{u} \\ \bar{v} \\ 1 \end{bmatrix} = K^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\alpha} & 0 & -\frac{u_0}{\alpha} \\ 0 & \frac{1}{\alpha} & -\frac{v_0}{\alpha} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{u-u_0}{\alpha} \\ \frac{v-v_0}{\alpha} \\ 1 \end{bmatrix} \quad \Rightarrow \lambda \begin{bmatrix} \bar{u} \\ \bar{v} \\ 1 \end{bmatrix} = \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$

Today's Outline

- Image Formation
- Other camera parameters
- Digital camera
- Perspective camera model
- Lens distortion

Radial Distortion



No distortion

Negative radial distortion
(Barrel distortion)
$k_1 < 0$

Positive radial distortion
(Pincushion distortion)
$k_1 > 0$

Radial Distortion in the OpenCV and Matlab Camera Models

- The standard model of radial distortion is a transformation **from the ideal** (**non-distorted**) coordinates (u, v) **to the real** (**distorted**) coordinates (u_d, v_d)

- For a given non distorted image point (u, v), the amount of distortion is a nonlinear function of its distance r from the principal point. For most lenses, **this simple quadratic model of radial distortion is sufficient**:

$$\begin{bmatrix} u_d \\ v_d \end{bmatrix} = (1 + k_1 r^2) \begin{bmatrix} u - u_0 \\ v - v_0 \end{bmatrix} + \begin{bmatrix} u_0 \\ v_0 \end{bmatrix}$$

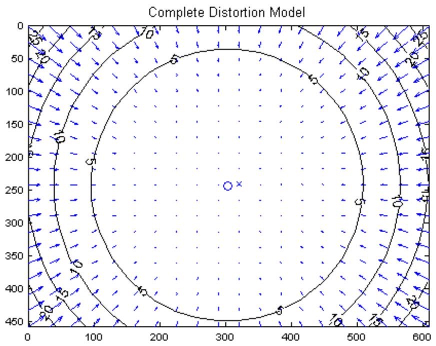
where $$r^2 = (u - u_0)^2 + (v - v_0)^2$$

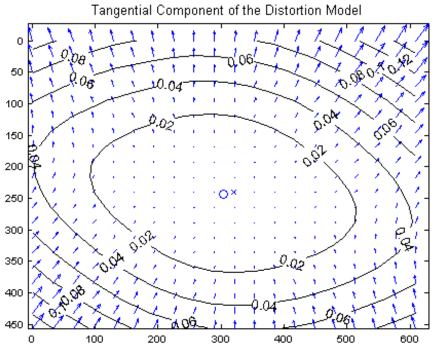
Radial & Tangential Distortion in the OpenCV and Matlab Camera Models

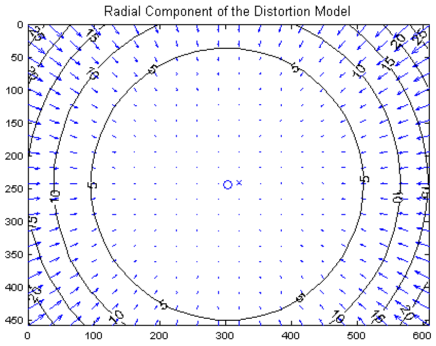
- **Radial Distortion:** Depending on the amount of distortion (an thus on the camera field of view), higher order terms can be introduced for the radial distortion
- **Tangential Distortion**: if the lens is misaligned (not perfectly parallel to the image sensor), a *non-radial* (tangential) distortion is introduced

$$\begin{bmatrix} u_d \\ v_d \end{bmatrix} = \overbrace{(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)\begin{bmatrix} u - u_0 \\ v - v_0 \end{bmatrix}}^{\text{Radial distortion}} + \overbrace{\begin{bmatrix} 2k_4(u-u_0)(v-v_0) + k_5(r^2 + 2(u-u_0)^2) \\ k_4(r^2 + 2(v-v_0)^2 + 2k_5(u-u_0)(v-v_0) \end{bmatrix}}^{\text{Tangential distortion}} + \begin{bmatrix} u_0 \\ v_0 \end{bmatrix}$$

This formula won't be asked at the exam ☺







The **left figure** shows the impact of the **complete distortion model (radial + tangential)** on each pixel of the image. Each arrow represents the effective displacement of a pixel induced by the lens distortion. Observe that points at the corners of the image are displaced by as much as **25 pixels**. The **center figure** shows the impact of the **tangential component of distortion**. On this plot, the **maximum induced displacement is 0.14 pixel** (at the upper left corner of the image). Finally, the **right figure** shows the impact of the **radial component of distortion**. This plot is very similar to the full distortion plot, showing that the tangential component could very well be discarded in the complete distortion model. On the three figures, the **cross** indicates the **center of the image** (i.e., the center of the two diagonals), and the **circle** the location of the **principal point**.

Summary: Perspective projection equations

- A world's 3D point $P_w = (X_w, Y_w, Z_w)$ projects into the image point $p = (u, v)$

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K[R|T] \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad \text{where} \quad K = \begin{bmatrix} \alpha & 0 & u_0 \\ 0 & \alpha & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

- To account for radial distortion, distorted pixel coordinates (u_d, v_d) can be obtained as:

$$\begin{bmatrix} u_d \\ v_d \end{bmatrix} = (1 + k_1 r^2) \begin{bmatrix} u - u_0 \\ v - v_0 \end{bmatrix} + \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} \quad \text{where} \quad r^2 = (u - u_0)^2 + (v - v_0)^2$$

- For convenience, the projection of P_w into pixel coordinates, including lens distortion, is usually denoted:

$$\begin{bmatrix} u_d \\ v_d \end{bmatrix} = \pi(P_W, K, k_1, R, T)$$

See also the OpenCV documentation (link)

Summary (things to remember)

- This-lens equation
- From the thin lens to the pinhole camera
- Perspective Projection Equation
- Vanishing points and lines
- Intrinsic and extrinsic parameters $(\boldsymbol{K}, \boldsymbol{R}, \boldsymbol{T})$
- Homogeneous coordinates
- Normalized image coordinates
- Radial distortion

Readings

- Chapter 4 of Autonomous Mobile Robots book: link
- Ch. 2.1 and 2.3 of Szeliski book, 2nd Edition

Understanding Check

Are you able to:

- Explain what a blur circle is?
- Derive the thin lens equation and perform the pinhole approximation?
- Explain how to build an Ames room?
- Derive a relation between the field of view and the focal length?
- Proof the perspective projection equation, including lens distortion and world-to-camera projection?
- Explain normalized image coordinates and their geometric explanation?
- Define vanishing points and lines?
- Prove that parallel lines intersect at vanishing points?