ROBOTICS & PERCEPTION GROUP

Universität Zürich[UZH]

Lecture 02
Image Formation

Prof. Dr. Davide Scaramuzza

sdavide@ifi.uzh.ch

Today's Class

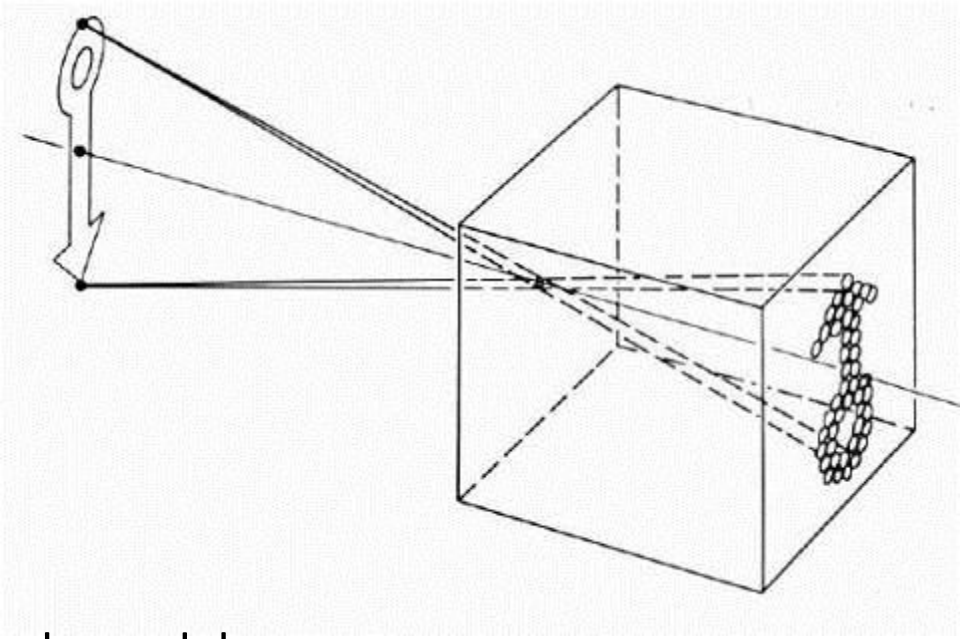
- Summary of the last lecture
- Perspective camera model
- Lens distortion
- Camera calibration
- List of mini-projects

Summary of the last lecture

The camera

Sony Cybershot WX1

Pinhole camera model



- Pinhole model:
  - Captures **beam of rays** – all rays through a single point
  - The point is called **Center of Projection** or **Optical Center**
  - The image is formed on the **Image Plane**

- We will use the pinhole camera model to describe how the image is formed

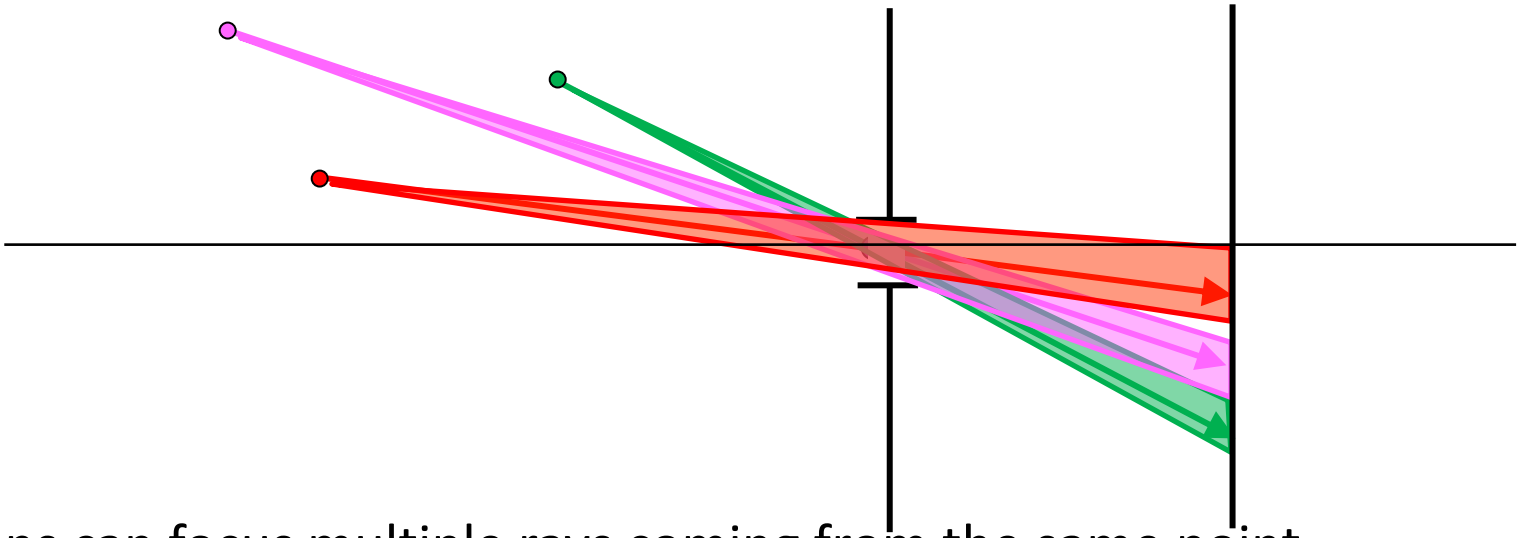
Why use a lens?

- *The ideal pinhole:*

  only one ray of light reaches each point on the film

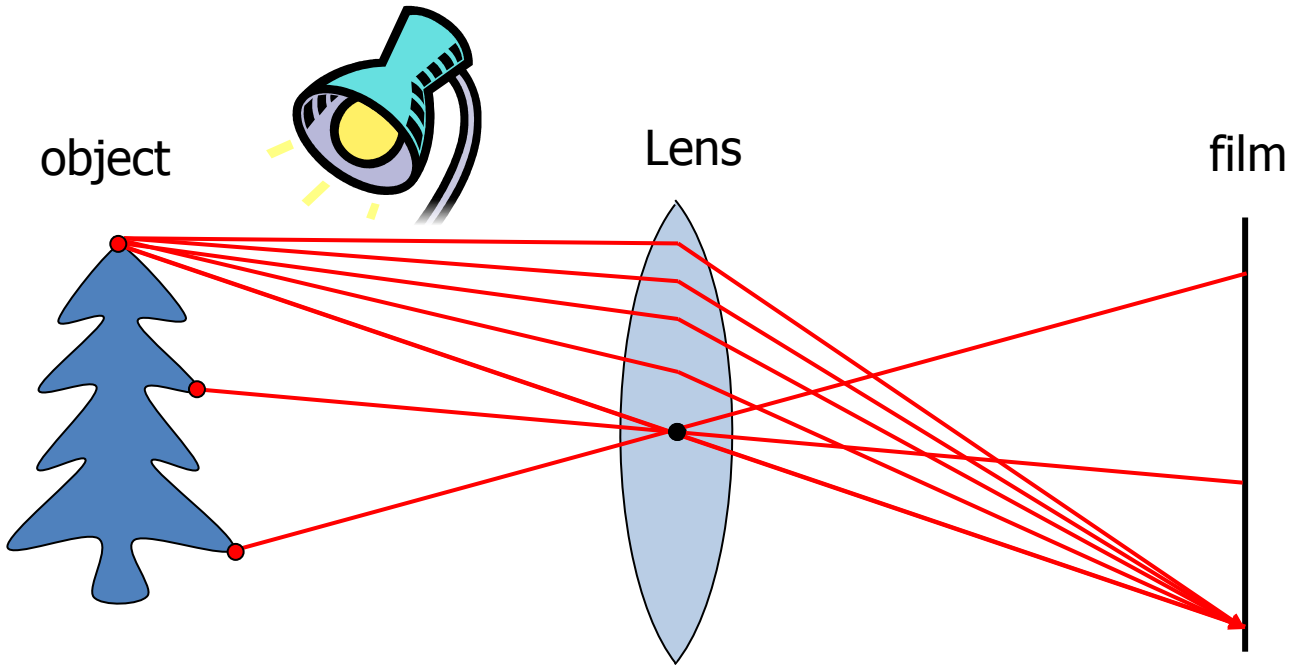
  ⇨ image can be very dim

- Making the pinhole bigger (i.e. aperture)…



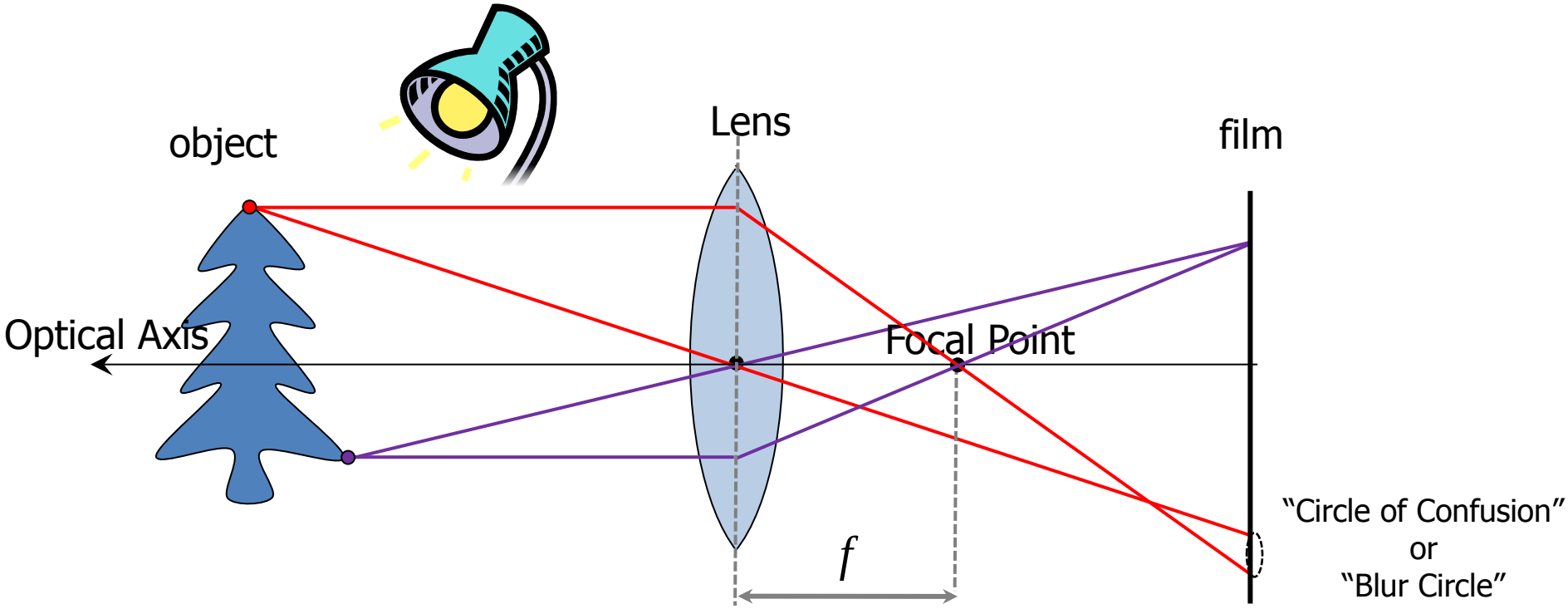
- A lens can focus multiple rays coming from the same point

Image formation using a converging lens



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

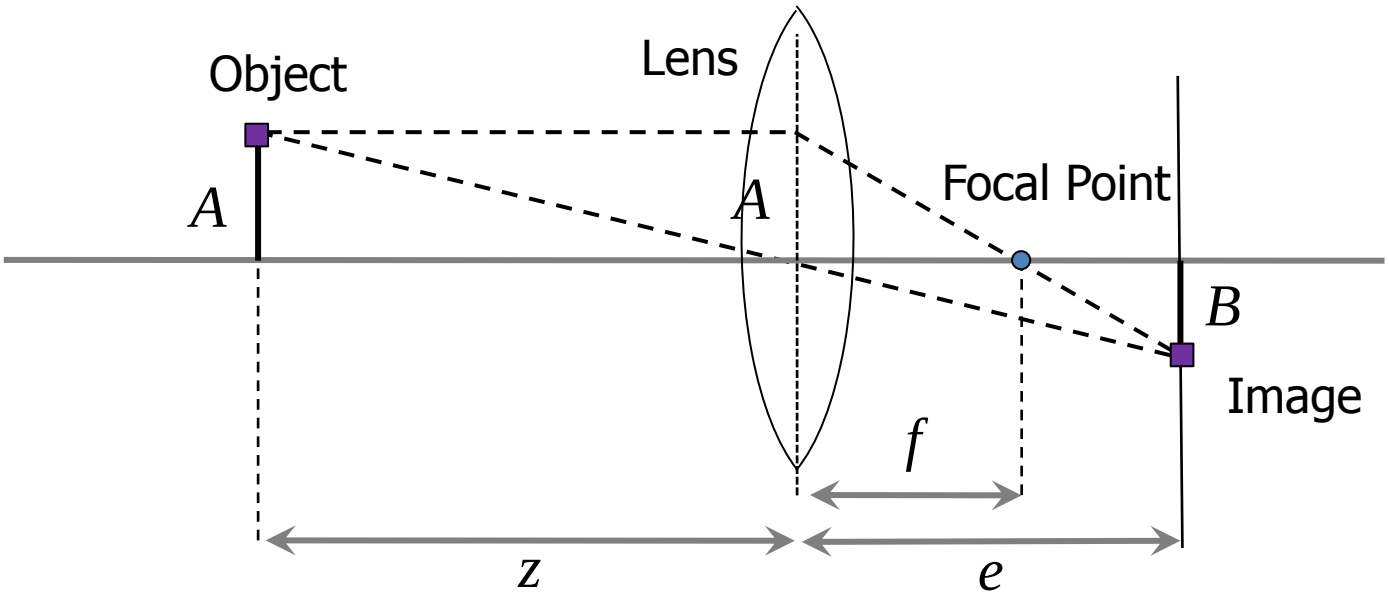
“In focus”



- There is a specific distance from the lens, at which world points are “in focus” in the image
- Other points project to a “blur circle” in the image

The Pin-hole approximation

- What happens if $z \gg f$?

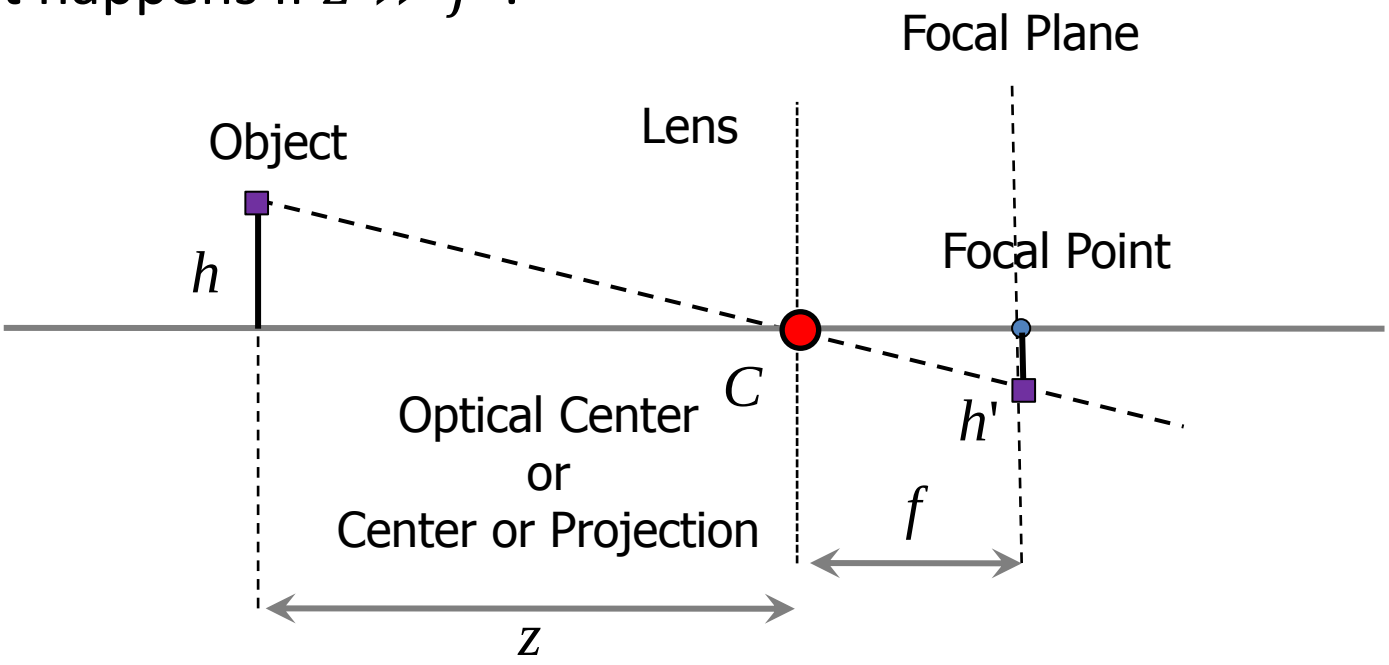


- We need to adjust the image plane such that objects at infinity are in focus

$$\frac{1}{f} = \underbrace{\frac{1}{z}}_{\cong 0} - \frac{1}{e} \quad \Rightarrow \frac{1}{f} \approx \frac{1}{e} \Rightarrow f \approx e$$

The Pin-hole approximation

- What happens if $z \gg f$?

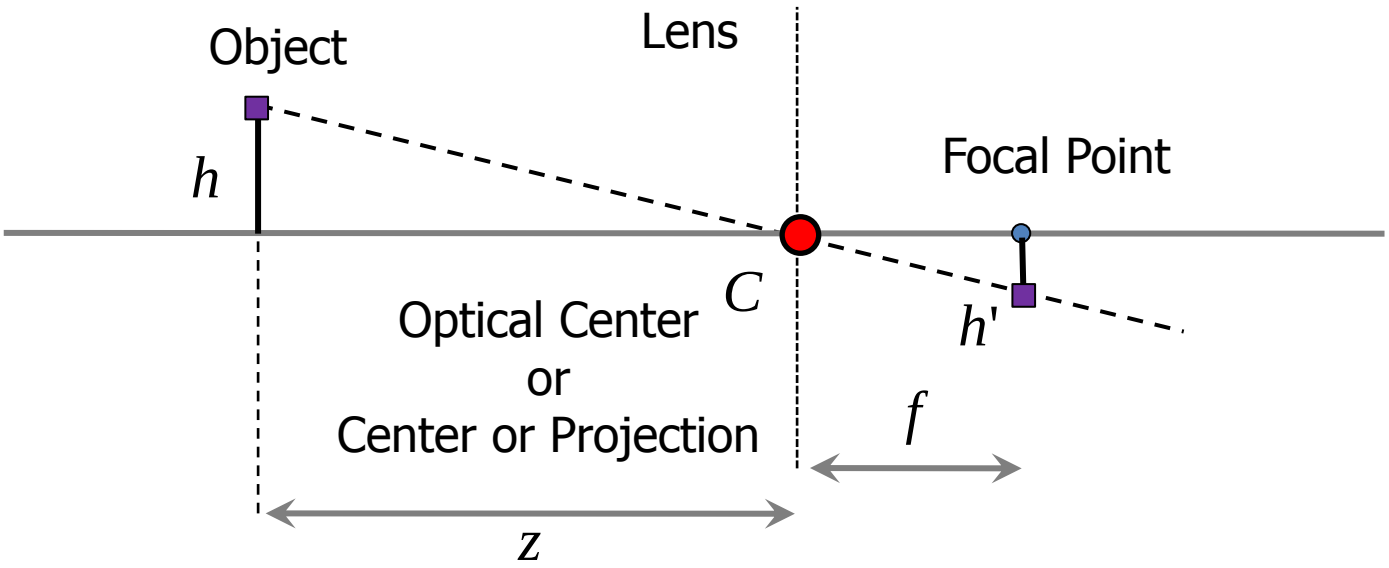


- We need to adjust the image plane such that objects at infinity are in focus

$$\frac{1}{f} = \underbrace{\frac{1}{z}}_{\cong 0} - \frac{1}{e} \quad \Rightarrow \frac{1}{f} \approx \frac{1}{e} \Rightarrow f \approx e$$

The Pin-hole approximation

- What happens if $z \gg f$?



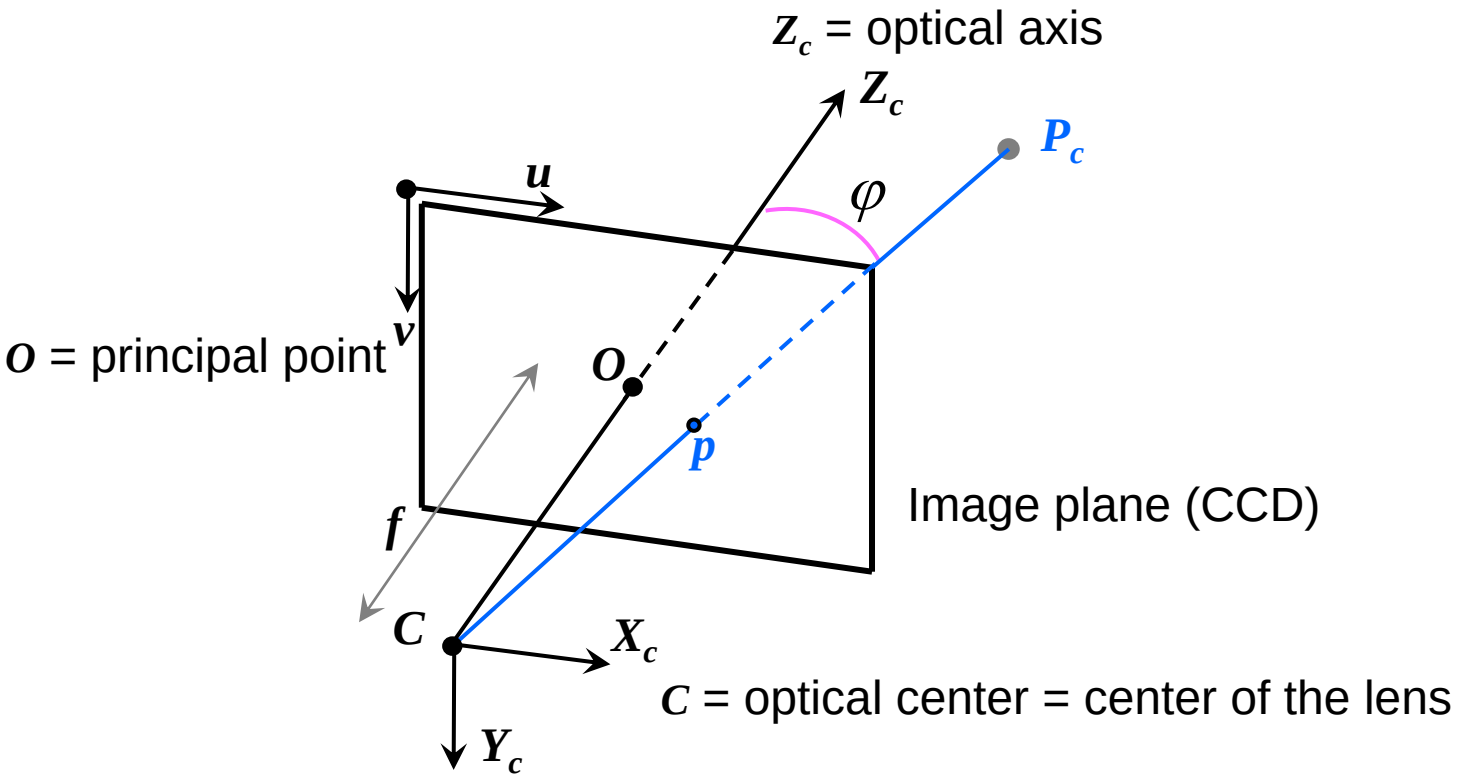
- We need to adjust the image plane such that objects at infinity are in focus

$$\frac{h'}{h} = \frac{f}{z} \Rightarrow h' = \frac{f}{z} h$$

- The dependence of the apparent size of an object on its depth (i.e. distance from the camera) is known as **perspective**

Perspective Projection

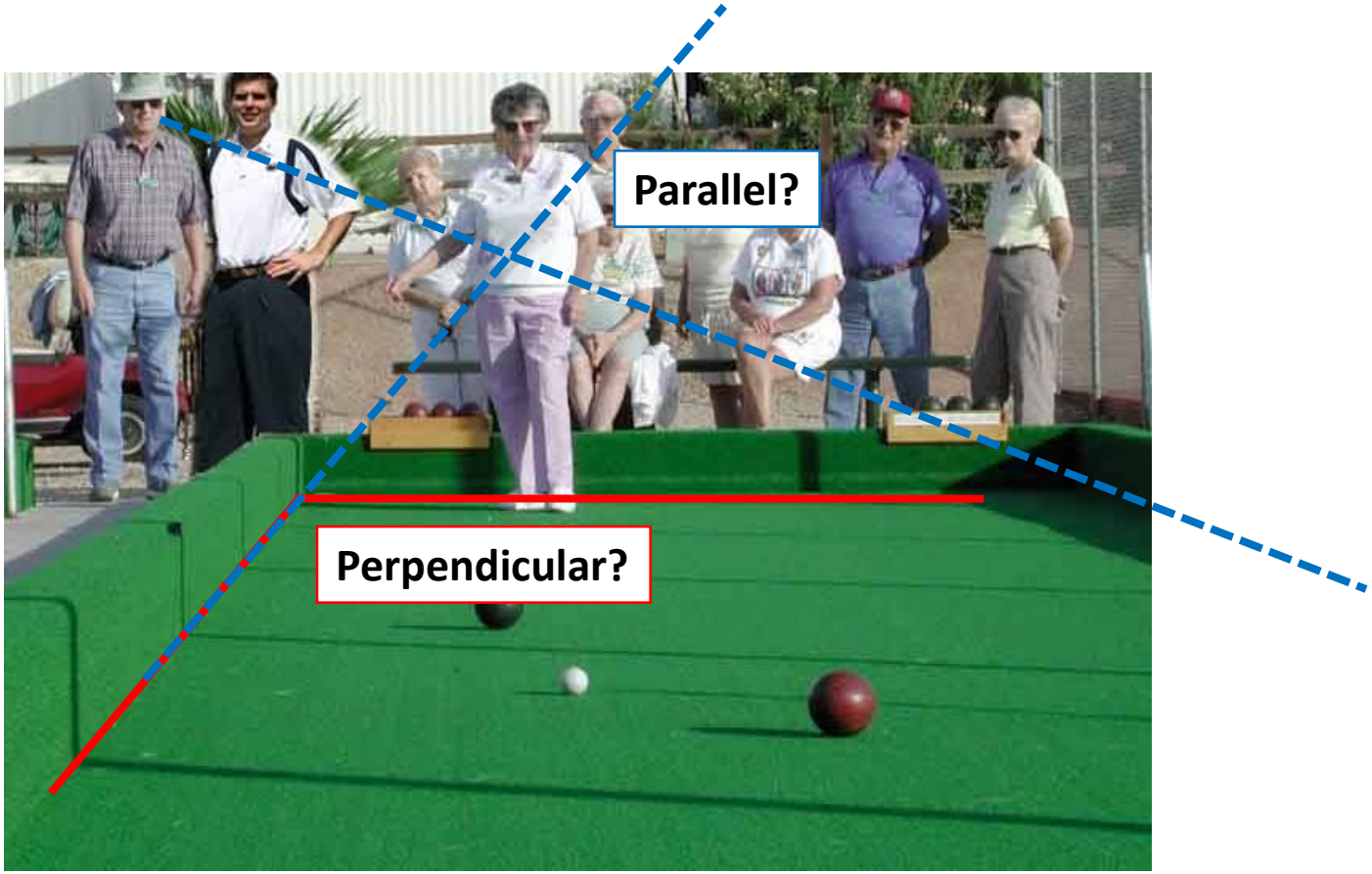
- For convenience, the image plane is usually represented in front of C such that the image preserves the same orientation (i.e. not flipped)
- A camera does not measure distances but angles!



Projective Geometry

What is lost?

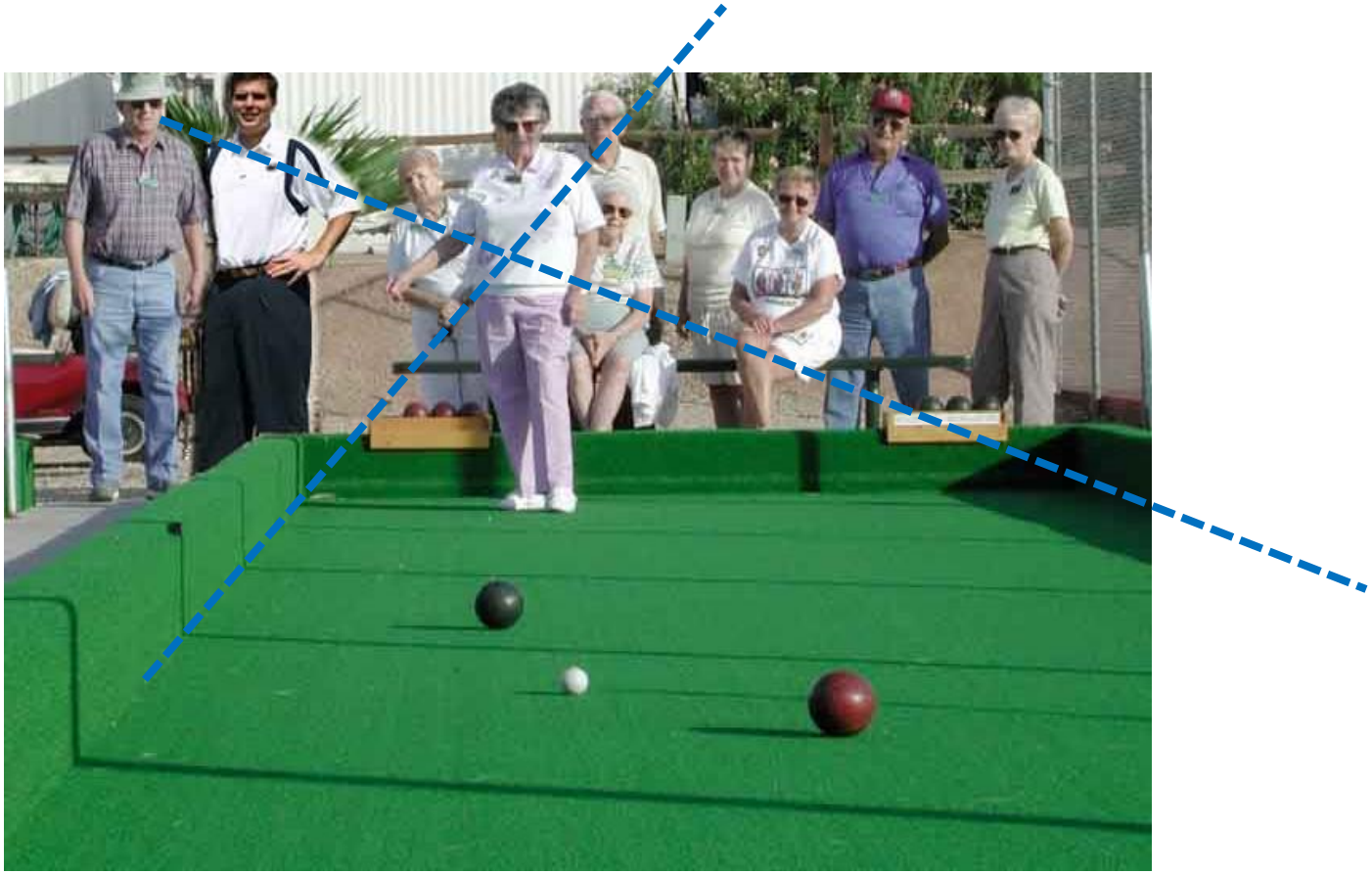
- Length
- Angles



Projective Geometry

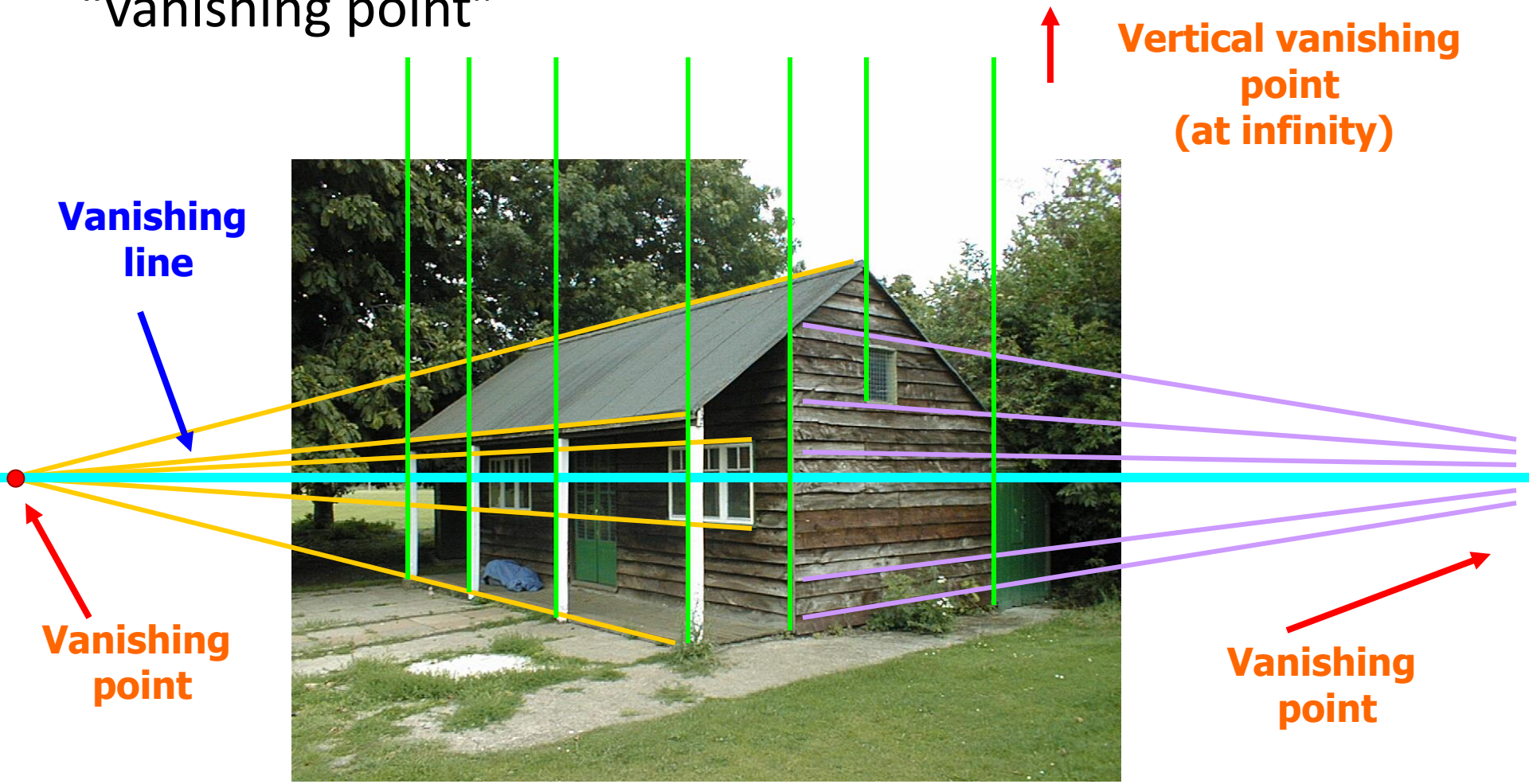
What is preserved?

- Straight lines are still straight



Vanishing points and lines

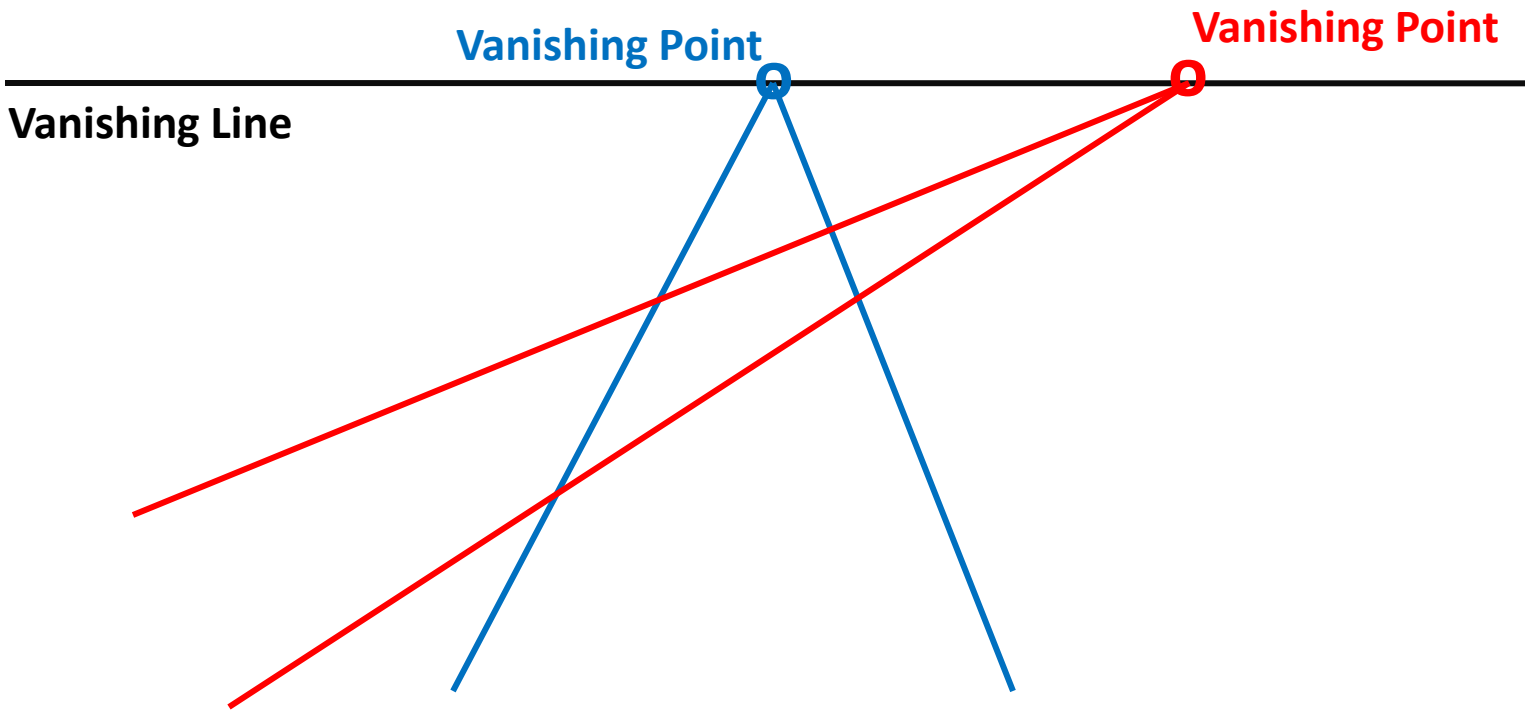
Parallel lines in the world intersect in the image at a "vanishing point"



Slide from Efros, Photo from Criminisi

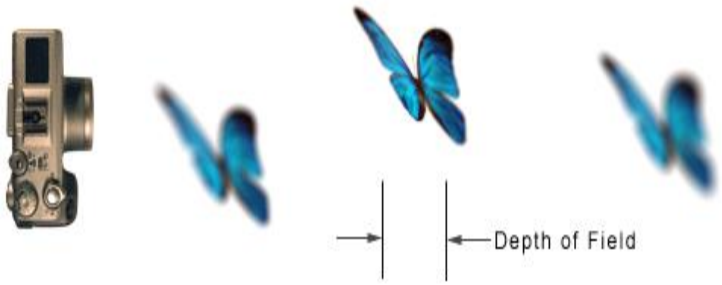
Vanishing points and lines

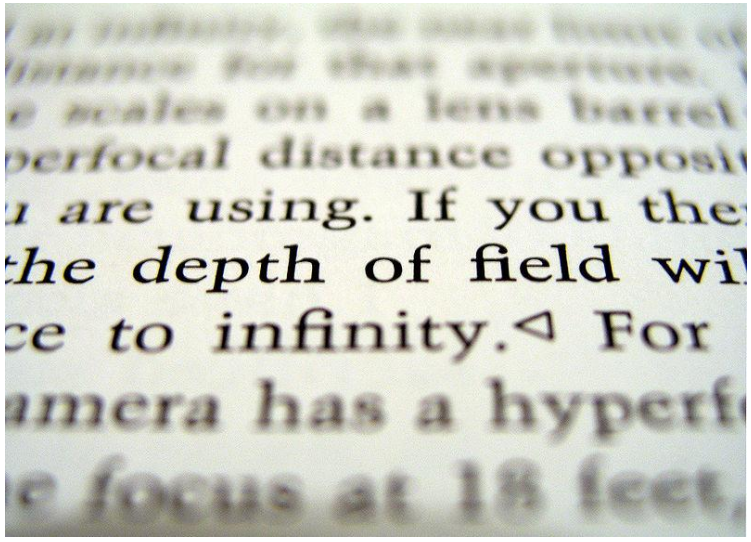
Parallel **planes** in the world intersect in the image at a "vanishing line"



Focus and depth of field

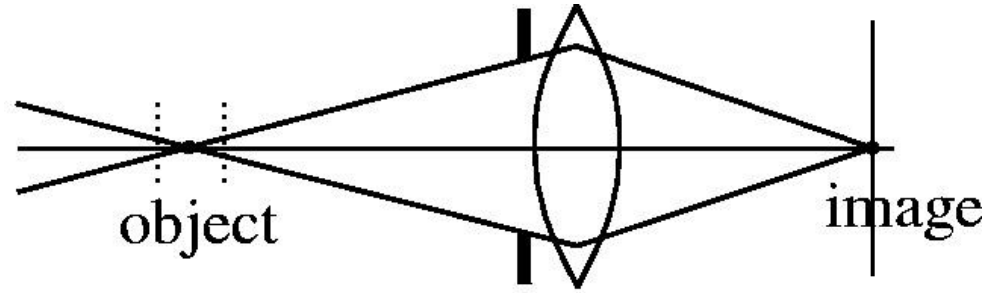
- Depth of field (DOF) is the distance between the nearest and farthest objects in a scene that appear acceptably sharp in an image.

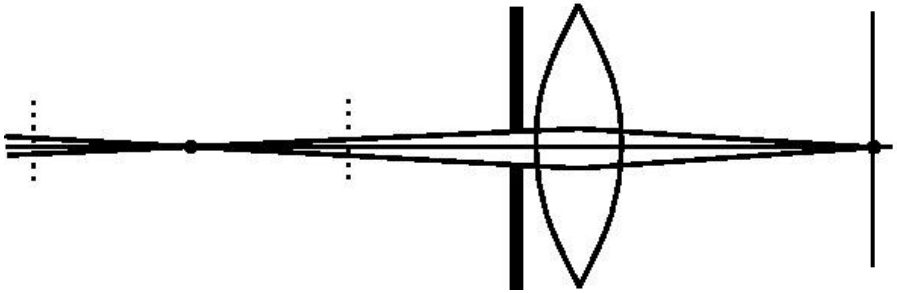




Focus and depth of field

- How does the aperture affect the depth of field?

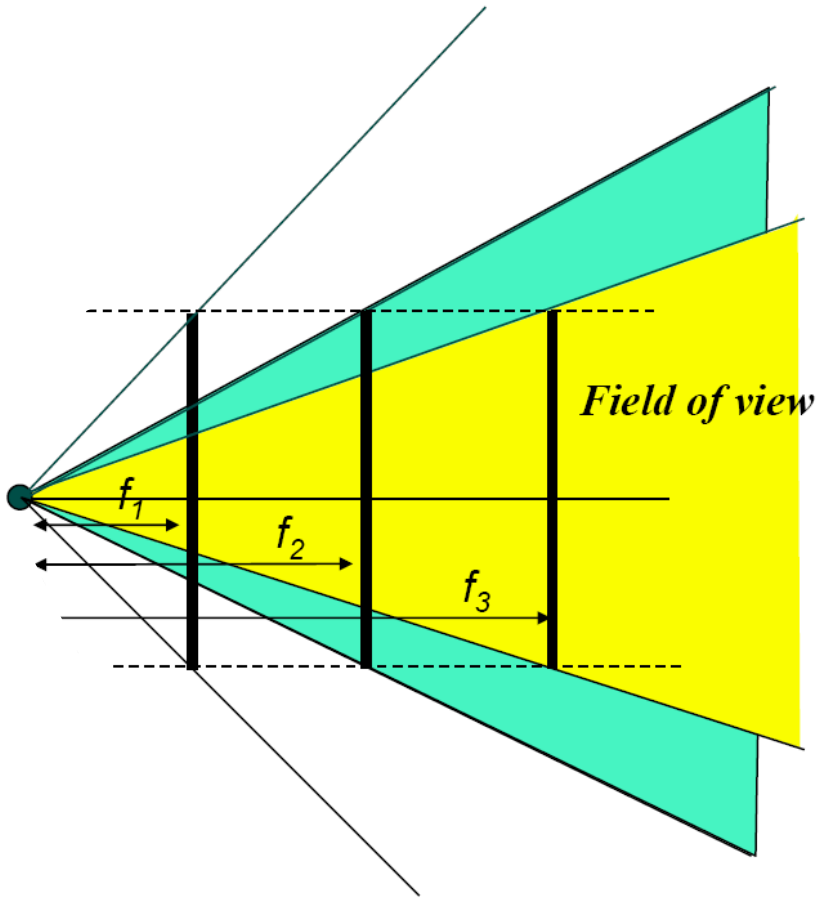




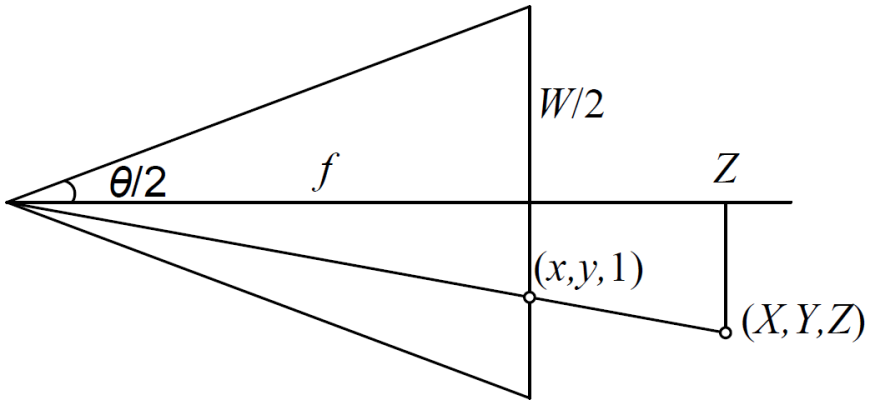
- A smaller aperture increases the range in which the object is approximately in focus

Field of view depends on focal length

- As ***f*** gets smaller, image becomes more *wide angle*
  - more world points project onto the finite image plane
- As ***f*** gets larger, image becomes more *narrow angle*
  - smaller part of the world projects onto the finite image plane



Field of view



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

Smaller FOV = larger Focal Length

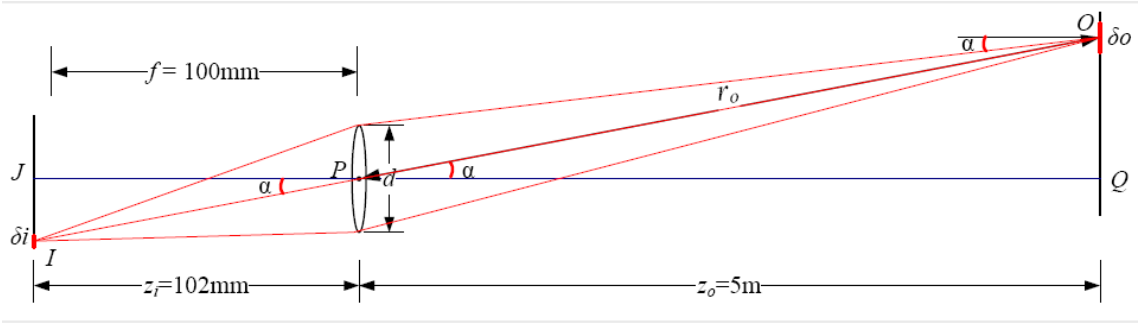
Vignetting

- Tendency of the brightness of the image to fall off towards the edge of the image

- Why and how can we remove it?

Vignetting

- “natural”: the light that reaches the patch on the image sensor is reduced by an amount that depends on angle α



- “mechanical”: occlusion of rays near the periphery of the lens elements in a compound lens

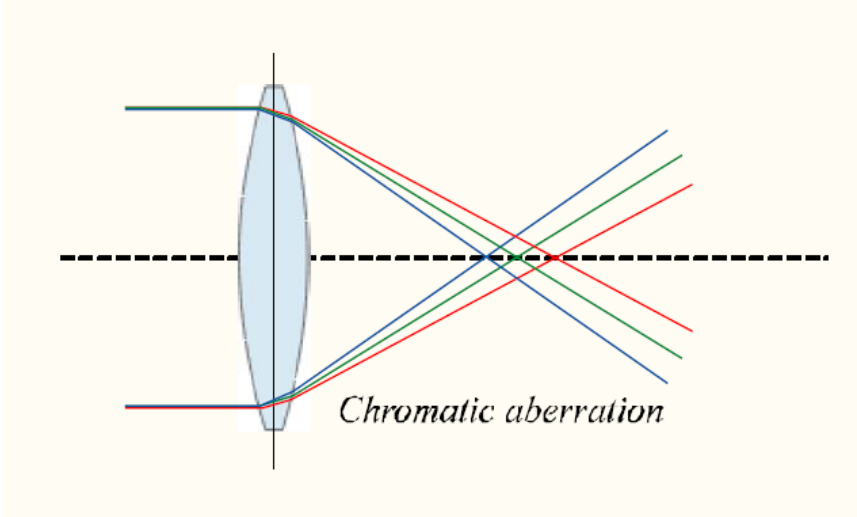
Chromatic aberration

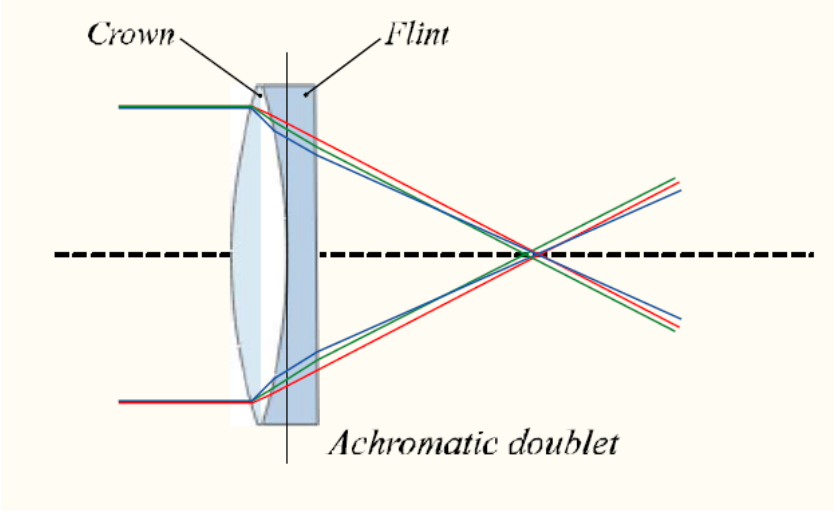
What causes it?

Chromatic aberration

- Because the index of refraction for glass varies slightly as a function of wavelength, light of different colors focuses at slightly different distances (and hence also with slightly different magnification factors)

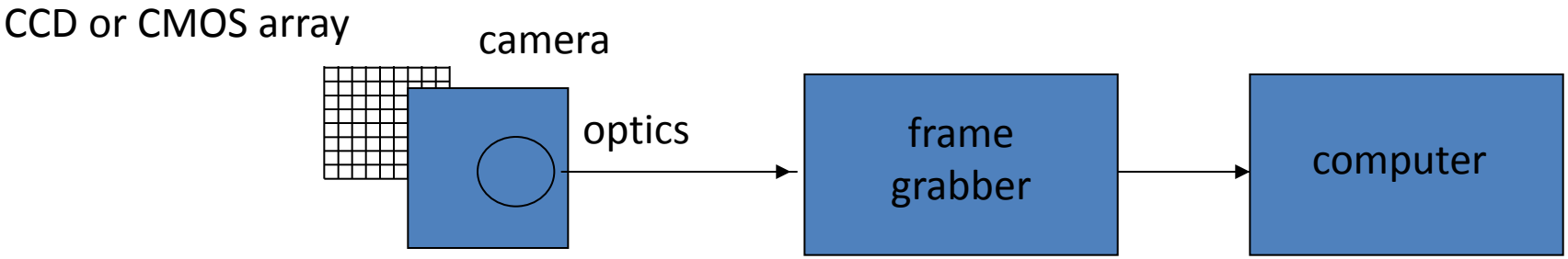
- In order to reduce chromatic aberration, most photographic lenses today are compound lenses made of different glass elements (with different coatings).

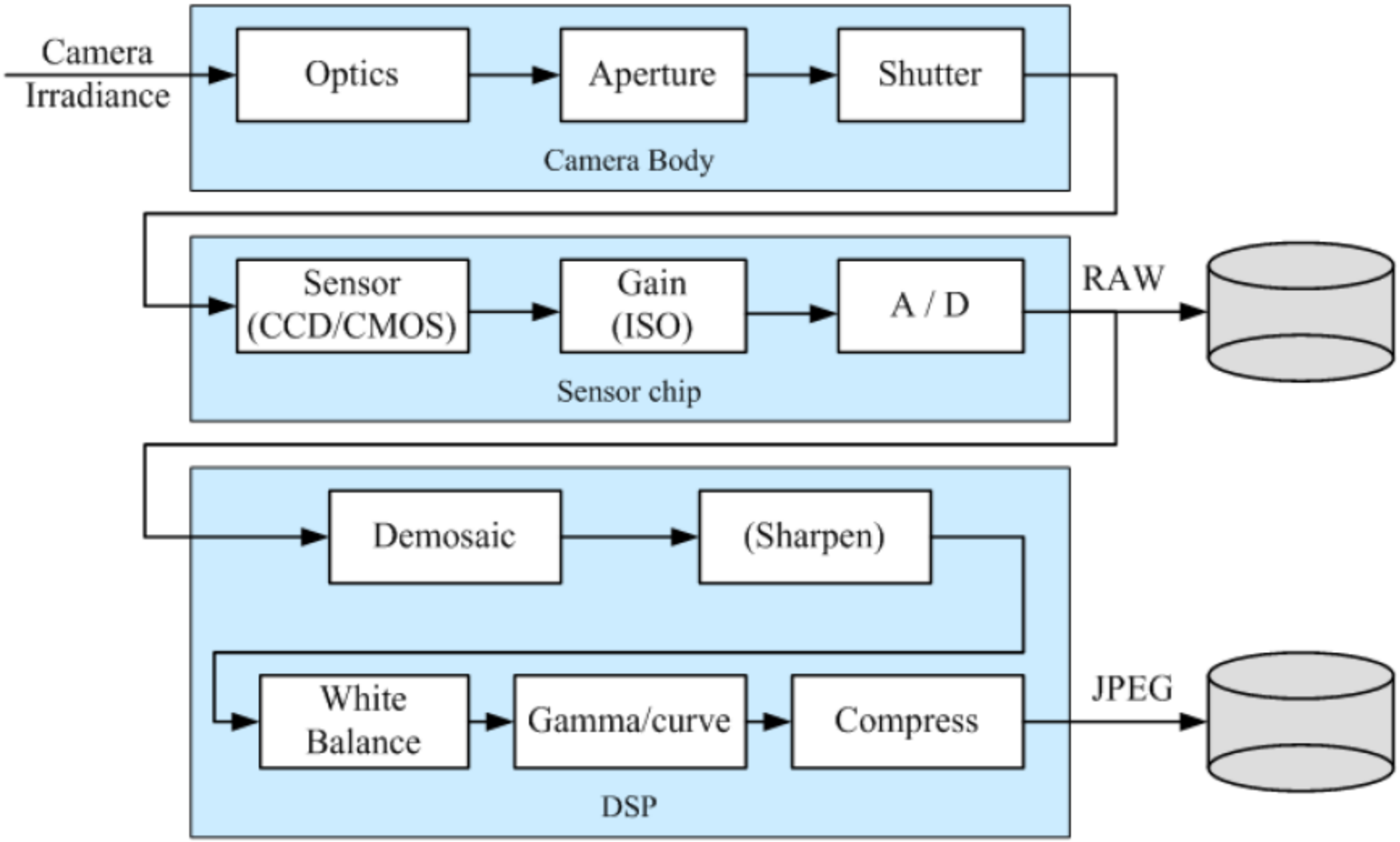




Digital cameras

- Film → sensor array
- Often an array of charge coupled devices
- Each CCD/CMOS is light sensitive diode that converts photons (light energy) to electrons




Camera
Irradiance
Optics
Aperture
Shutter
Camera Body
Sensor
(CCD/CMOS)
Gain
(ISO)
A / D
Sensor chip
RAW
Demosaic
(Sharpen)
White
Balance
Gamma/curve
Compress
DSP
JPEG

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

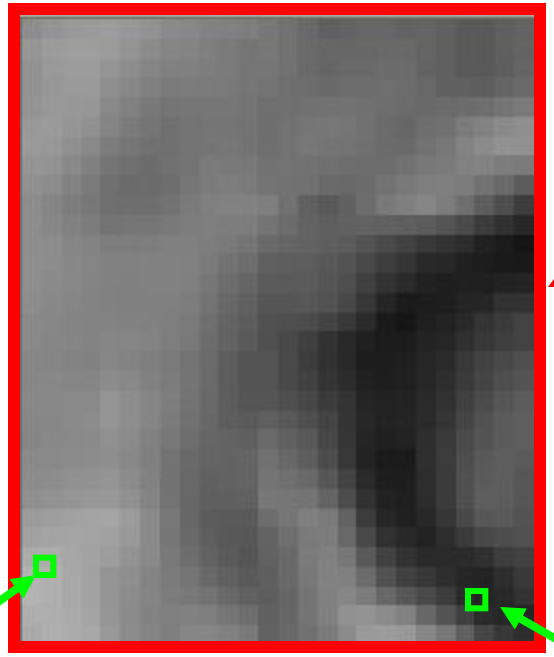
Sample: **mvBlueFOX-IGC200wG** means version with housing and 752 x 480 CMOS gray scale sensor.
mvBlueFOX-MLC200wG means single-board version without housing and with 752 x 480 CMOS gray scale sensor.

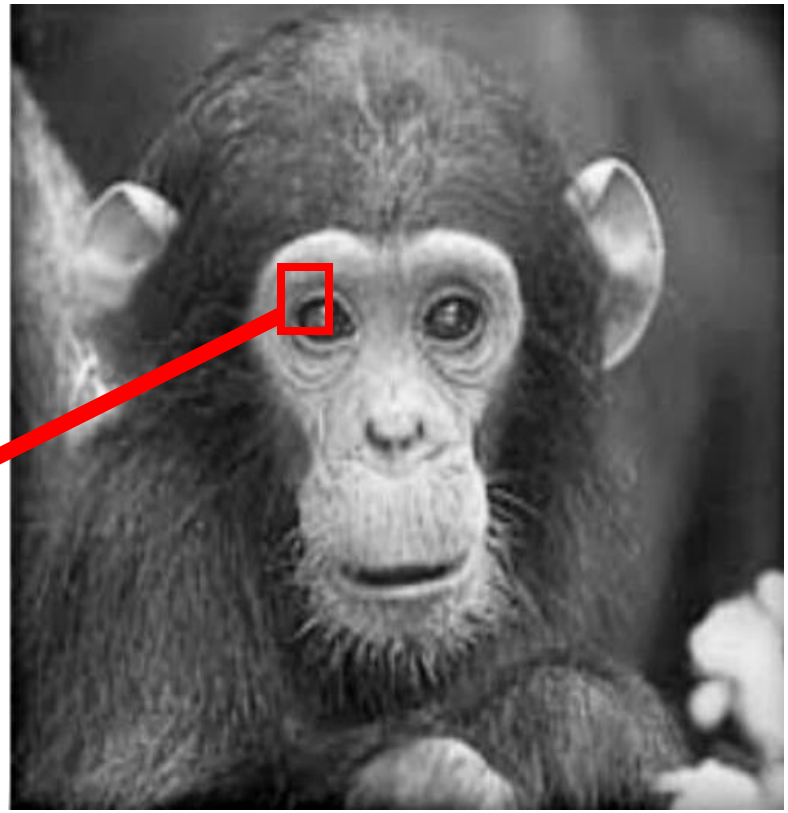
Hardware Features

Gray scale / Color		Gray scale (G) / Color (C)
Interface		USB 2.0 (up to 480 Mbit/s)
Image formats		Mono8, Mono10, BayerGR8, BayerGR10
Triggers		External hardware based (optional), software based (depending on the sensor) or free run
Size w/o lens (W x H x L) | Weight w/o lens	mvBlueFOX-IGC: mvBlueFOX-MLC:	39.8 x 39.8 x 16.5 mm | approx. 10 g 35 x 33 x 25 mm (without lens mount) | approx. 80 g
Permissible ambient temperature	Operation: Storage:	0 .. 45 °C / 30 to 80 % RH -20 .. 60 °C / 20 to 90 % RH
Lens mounts		Back focus adjustable C/CS-mount lens holder / C-mount, CS-mount or optional S-mount
Digital I/Os	mvBlueFOX-IGC (optional) mvBlueFOX-MLC	1 / 1 opto-isolated 1 / 1 opto-isolated or 2 / 2 TTL compliant
Conformity		CE, FCC, RoHS
Driver		mvIMPACT Acquire SDK
Operating systems		Windows®, Linux® - 32 bit and 64 bit
Special features		Micro-PLC, automatic gain / exposure control, binning, screw lock connectors

Digital images

Pixel Intensity : [0,255] (8 bits)



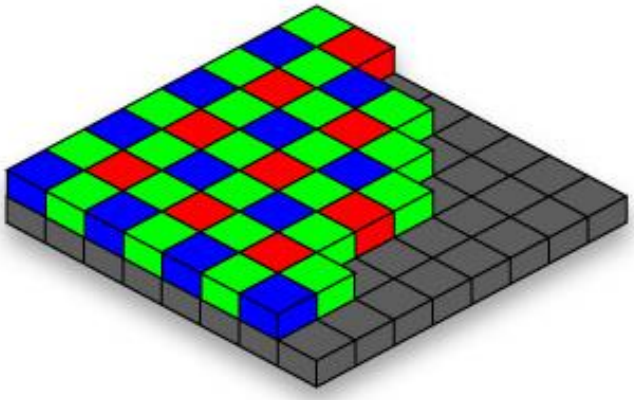


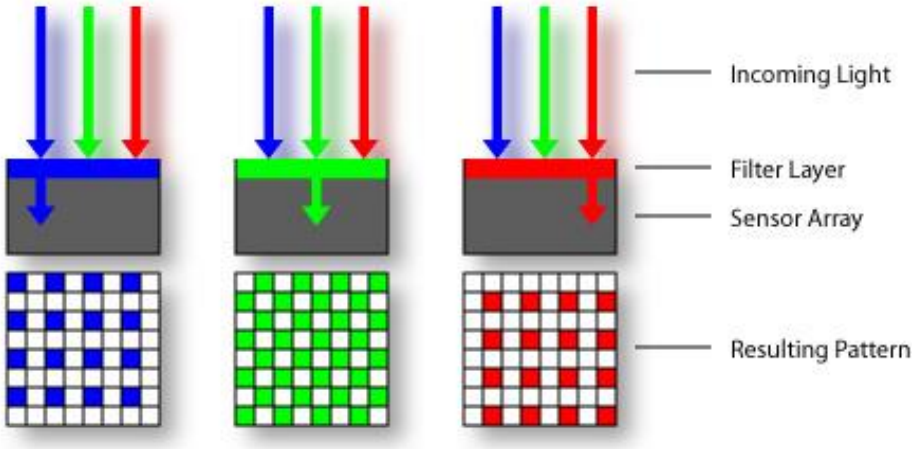
im[176][201] has value 164

im[194][203] has value 37

Color sensing in digital cameras

Bayer grid

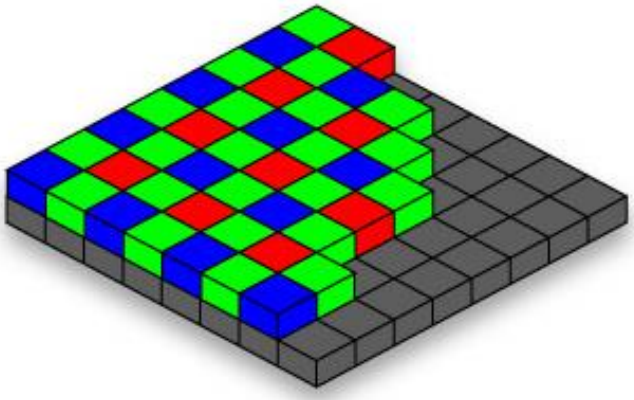




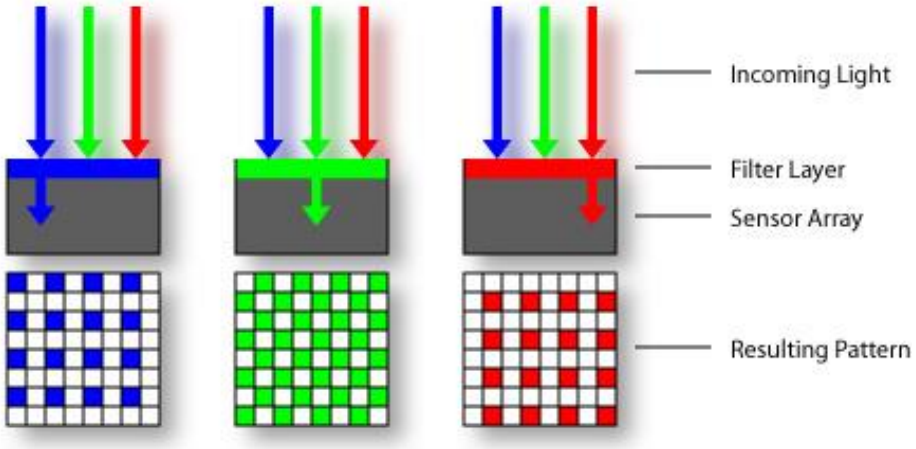
- The Bayer pattern (Bayer 1976) places green filters over half of the sensors (in a checkerboard pattern), and red and blue filters over the remaining ones.
- This is because the luminance signal is mostly determined by green values and the visual system is much more sensitive to high frequency detail in luminance than in chrominance.

Color sensing in digital cameras

Bayer grid



Estimate missing components from neighboring values (demosaicing)



A newer chip design by Foveon (http://www.foveon.com) stacks the red, green, and blue sensors beneath each other, but it has not yet gained widespread adoption.

Color images:

RGB color space

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

R

G

B

Outline of this lecture

- Perspective camera model
- Lens distortion
- Camera calibration
  - DLT algorithm

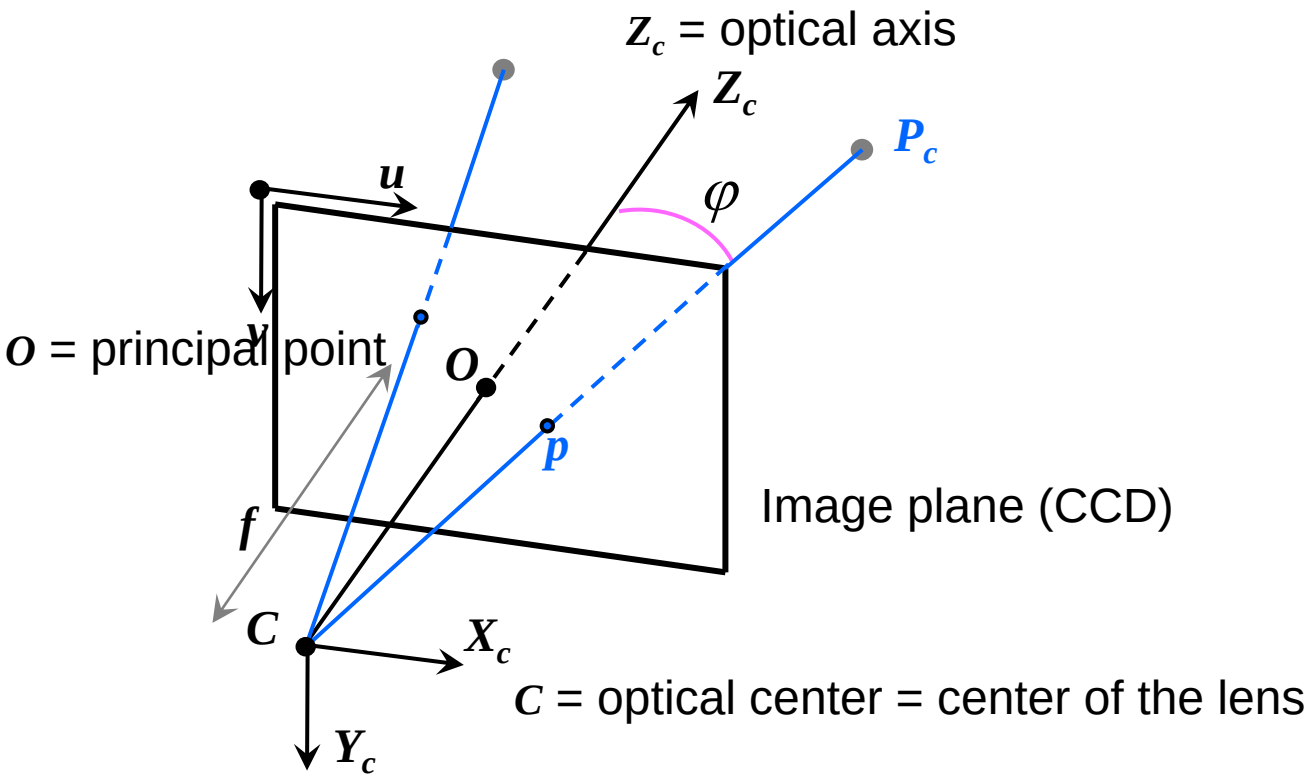
Perspective and art

- Use of correct perspective projection indicated in 1st century B.C. frescoes
- Skill resurfaces in Renaissance: artists develop systematic methods to determine perspective projection (around 1480-1515)

Raphael

Durer, 1525

Perspective Camera



- For convenience, the image plane is usually represented in front of C such that the image preserves the same orientation (i.e. not flipped)

- Note: **a camera does not measure distances but angles!**
  ⇨ a camera is a “bearing sensor”

From World to Pixel coordinates

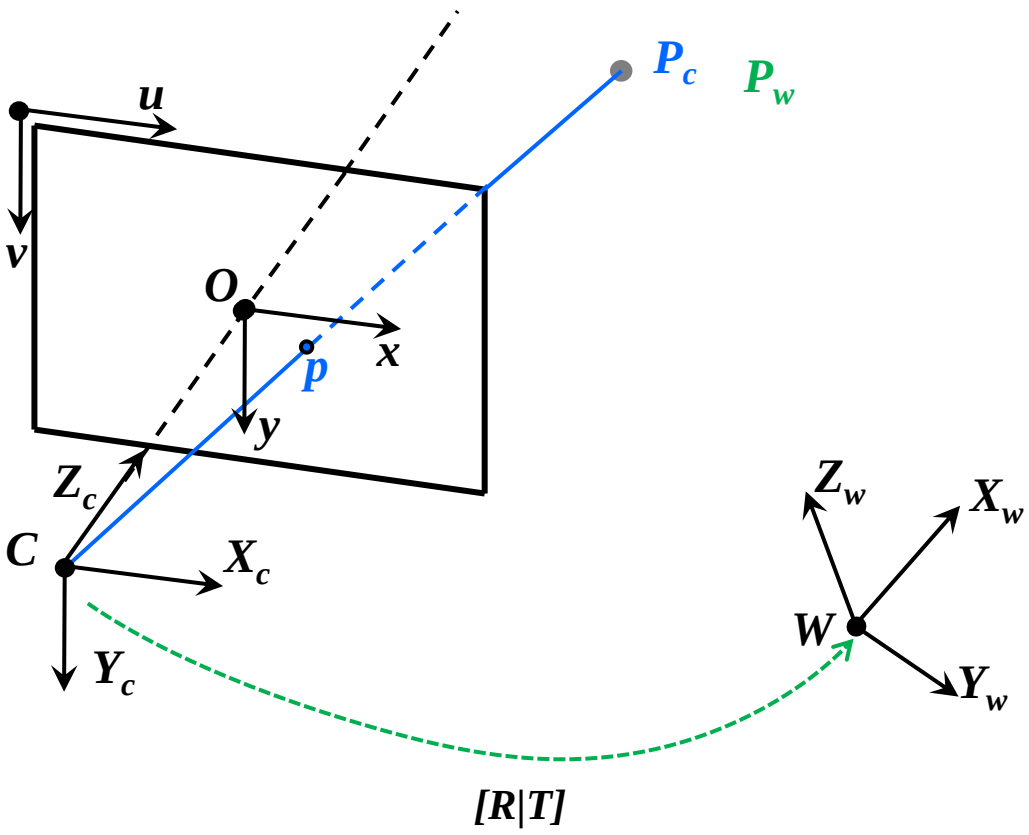
Find pixel coordinates (u,v) of point P_w in the world frame:

0. Convert world point P_w to camera point P_c

Find pixel coordinates (u,v) of point P_c in the camera frame:

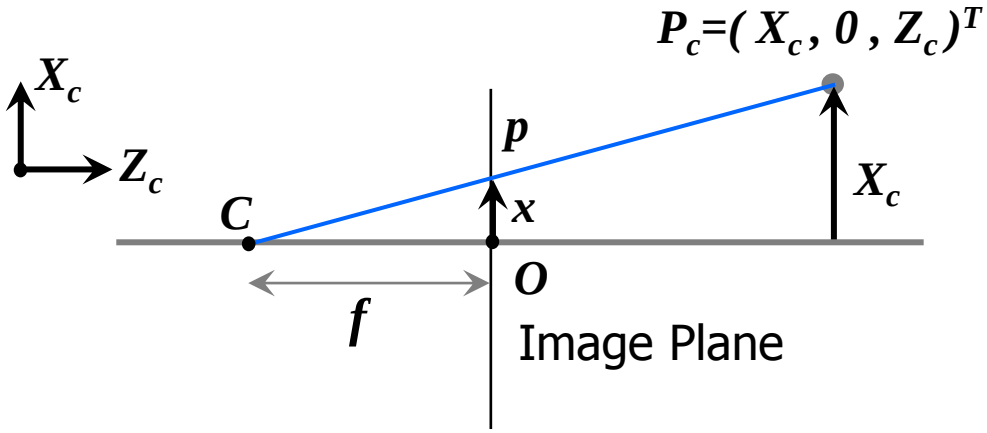
1. Convert P_c to image-plane coordinates (x,y)

2. Convert P_c to (discretised) pixel coordinates (u,v)



Perspective Projection (1)

From the Camera frame to the image plane



- The Camera point $P_c=(X_c, 0, Z_c)^T$ 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, in the general case:

$$\frac{y}{f} = \frac{Y_c}{Z_c} \Rightarrow y = \frac{fY_c}{Z_c}$$

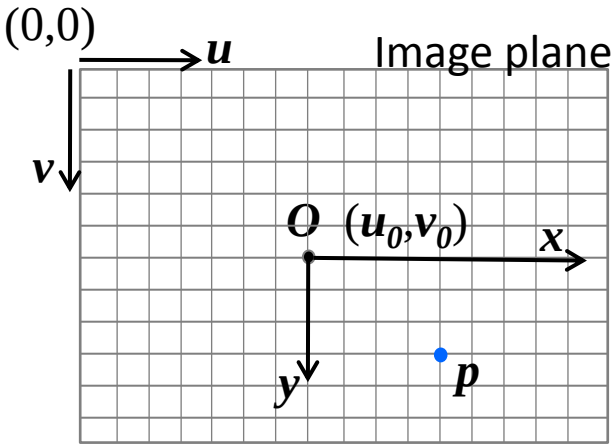
1. Convert P_c to image-plane coordinates (x,y)

2. Convert P_c to (discretised) pixel coordinates (u,v)

Perspective Projection (2)

From the Camera frame to pixel coordinates

- To convert ***p*** from the local image plane coords ***(x,y)*** to the pixel coords ***(u,v)***, we need to account for:
  - the pixel coords of the camera optical center $O = (u_0, v_0)$
  - Scale factors k_u, k_v for the pixel-size in both dimensions



So:

$$u = u_0 + k_u x \Rightarrow u = u_0 + \frac{k_u f X_c}{Z_c}$$

$$v = v_0 + k_v y \Rightarrow v = v_0 + \frac{k_v f Y_c}{Z_c}$$

- Use **Homogeneous Coordinates** for linear mapping from 3D to 2D, by introducing an extra element (scale):

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

Perspective Projection (3)

- So:

$$u = u_0 + \frac{k_u f X_c}{Z_c}$$

$$v = v_0 + \frac{k_v f Y_c}{Z_c}$$

Expressed in matrix form and homogenerous coordinates:

$$\begin{bmatrix} \lambda u \\ \lambda v \\ \lambda \end{bmatrix} = \begin{bmatrix} k_u f & 0 & u_0 \\ 0 & k_v f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$

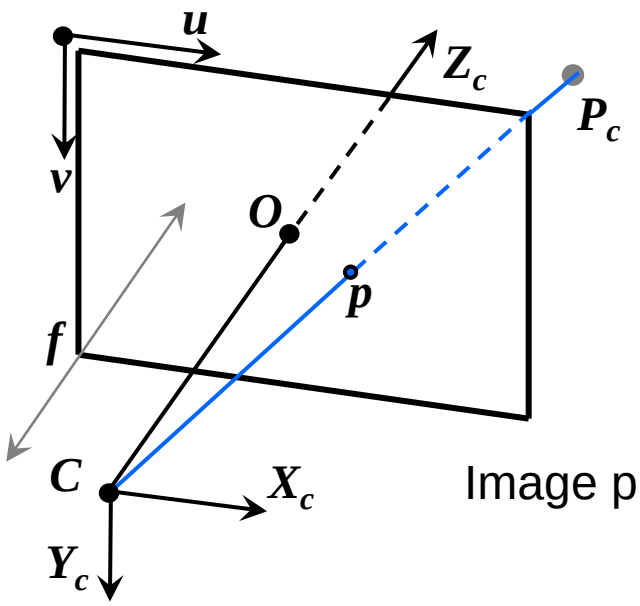


Image plane (CCD)

Or alternatively

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

Focal length in pixels

K is called “Calibration matrix” or “Matrix of Intrinsic Parameters”

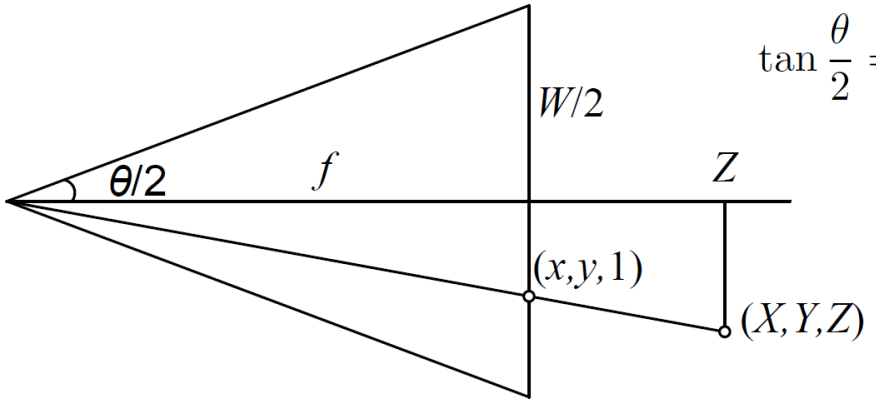
Sometimes, it is common to assume a skew factor ($K_{12} \neq 0$) to account for possible misalignments between CCD and lens. However, the camera manufacturing process today is so good that we can safely assume $K_{12} = 0$ and $\alpha_u = \alpha_v$.

Exercise

- Determine the Intrinsic Parameter Matrix (K) for a digital camera with image size 640×480 pixels and horizontal field of view equal to $90°$
- Assume the principal point in the center of the image and squared pixels
- What is the vertical field of view?

Exercise 1

- Determine the Intrinsic Parameter Matrix (K) for a digital camera with image size 640×480 pixels and horizontal field of view equal to $90°$
- Assume the principal point in the center of the image and squared pixels



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

$$f = \frac{640}{2\tan\frac{\theta}{2}} = 320\ \boldsymbol{pixels}$$

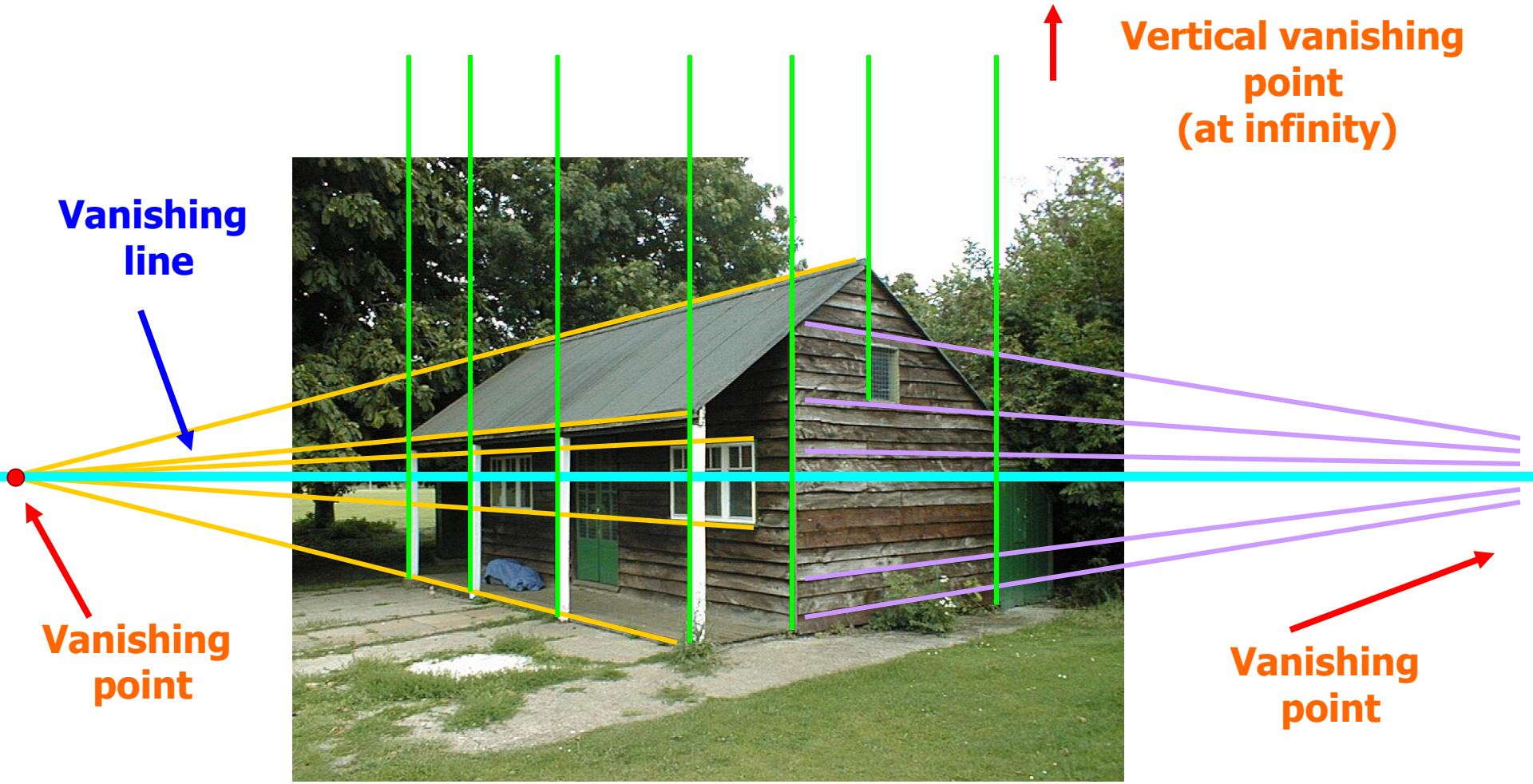
$$K = \begin{bmatrix} 320 & 0 & 320 \\ 0 & 320 & 240 \\ 0 & 0 & 1 \end{bmatrix}$$

- What is the vertical field of view?

$$\theta = 2\tan^{-1}\frac{H}{2f} = 2\tan^{-1}\frac{480}{2\cdot 320} = 73.74°$$

Exercise 2

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



Slide from Efros, Photo from Criminisi

Exercise 2

- Prove that world's parallel lines intersect at a vanishing point in the camera image
- Let's consider the perspective projection equation in standard coordinates:

$$u = u_0 + \alpha \frac{X}{Z}$$

$$v = v_0 + \alpha \frac{Y}{Z}$$

- Let's parameterize a 3D line with:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \end{bmatrix} + k \begin{bmatrix} l \\ m \\ n \end{bmatrix}$$

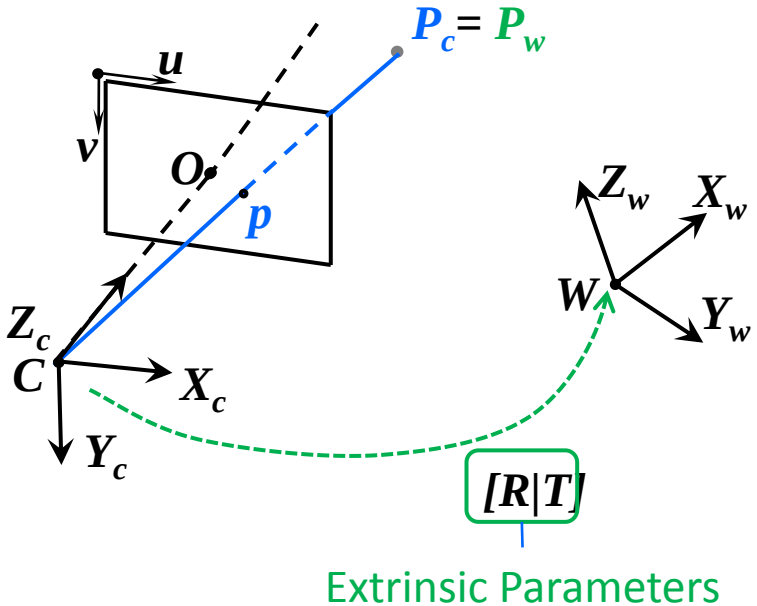
- Now substitute this into the camera perspective projection equation and compute the limit for $k \to \infty$
- What is the intuitive interpretation of this?

Perspective Projection (4)

From the Camera frame to the World frame

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

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



Projection Matrix (M)

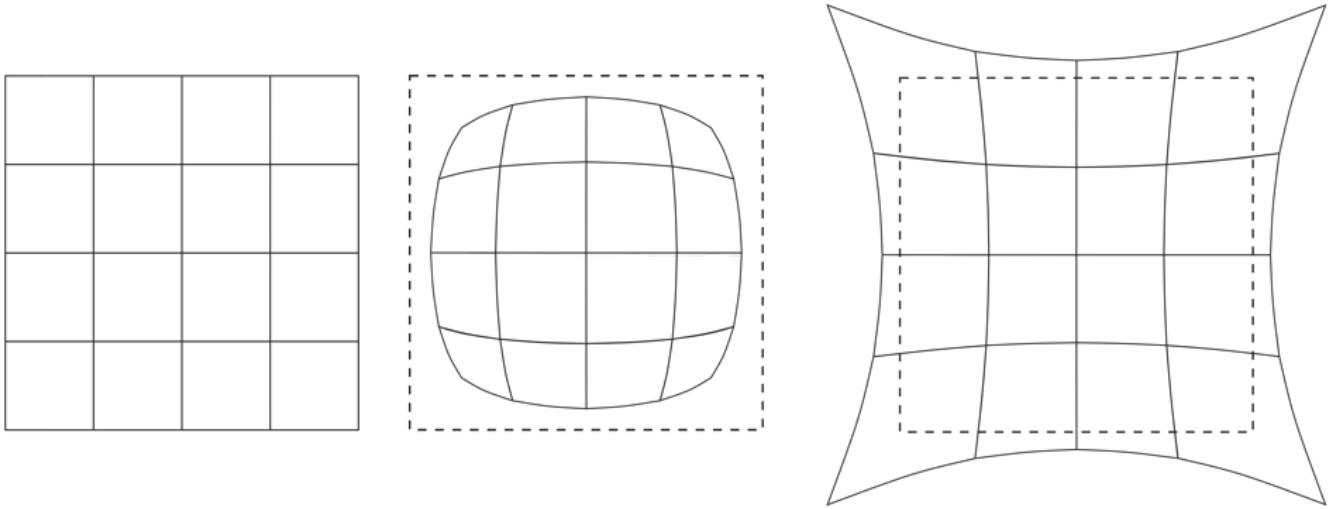
$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} \qquad \lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K[R|T] \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

Perspective Projection Equation

Outline of this lecture

- Perspective camera model
- Lens distortion
- Camera calibration
  - DLT algorithm

Radial Distortion



No distortion

Barrel distortion

Pincushion

Radial Distortion

- The standard model of radial distortion is a transformation from the ideal coordinates (u, v) (i.e., undistorted) to the real observable coordinates (distorted) (u_d, v_d)

- The amount of distortion of the coordinates of the observed image is a nonlinear function of their radial distance . For most lenses, a simple quadratic model of distortion produces good results

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

where

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

Summary: Perspective projection equations

- To recap, a 3D world point $P = (X_w, Y_w, Z_w)$ projects into the image point $p = (u, v)$

$$\tilde{p} = \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K\begin{bmatrix} R \mid T \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad \text{where} \quad K = \begin{bmatrix} \alpha & 0 & u_0 \\ 0 & \alpha & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

and λ is the depth ($\lambda = Z_C$) of the scene point

- If we want to take into account the radial distortion, then the distorted coordinates (u_d, v_d) (in pixels) can be obtained as

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

where

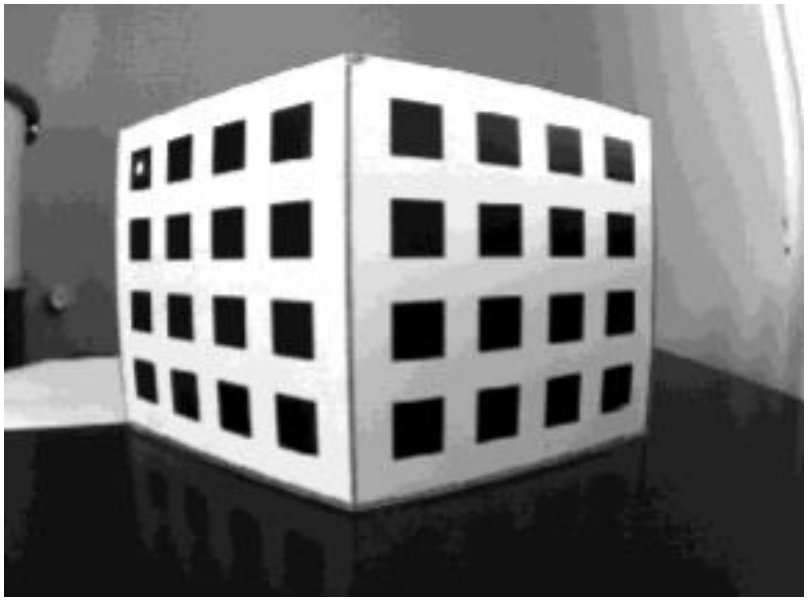
$$r^2 = (u - u_0)^2 + (v - v_0)^2$$

Outline of this lecture

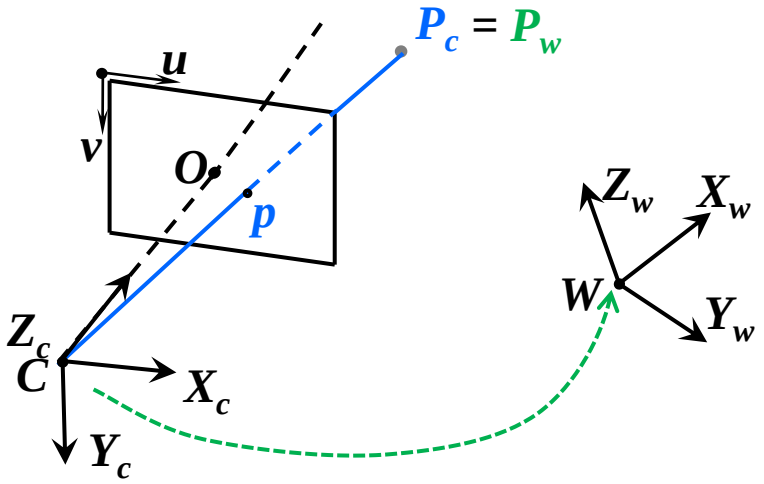
- Perspective camera model
- Lens distortion
- Camera calibration
  - DLT algorithm

Camera calibration

- Calibration is the process to determine the **intrinsic and extrinsic** parameters of the camera model
- A method proposed in 1987 by Tsai consists of measuring the 3D position of $n \geq 6$ control points on a three-dimensional calibration target and the 2D coordinates of their projection in the image. This problem is also called "**Resection**", or "**Perspective from n Points**", or "**Camera pose from 3D-to-2D correspondences**", and is one of the most widely used algorithms in Computer Vision and Robotics
- Solution: The intrinsic and extrinsic parameters are computed directly from the perspective projection equation; let's see how!



3D position of control points is assigned in a reference frame specified by the user



Camera calibration: Direct Linear Transform (DLT)

Our goal is to compute K, R, and T, that satisfy the perspective projection equation (we neglect the radial distortion)

$$\tilde{p} = \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K\begin{bmatrix} R \mid T \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \Rightarrow$$

$$\Rightarrow \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} \alpha_u r_{11} + u_0 r_{31} & \alpha_u r_{12} + u_0 r_{32} & \alpha_u r_{13} + u_0 r_{33} & \alpha_u t_1 + u_0 t_3 \\ \alpha_v r_{21} + v_0 r_{31} & \alpha_v r_{22} + v_0 r_{32} & \alpha_v r_{23} + v_0 r_{33} & \alpha_v t_2 + v_0 t_3 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

Camera calibration: Direct Linear Transform (DLT)

Our goal is to compute K, R, and T, that satisfy the perspective projection equation (we neglect the radial distortion)

$$\tilde{p} = \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K\begin{bmatrix} R \mid T \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \Rightarrow$$

$$\Rightarrow \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

Camera calibration: Direct Linear Transform (DLT)

Our goal is to compute K, R, and T, that satisfy the perspective projection equation (we neglect the radial distortion)

$$\Rightarrow \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = M \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} m_1^T \\ m_2^T \\ m_3^T \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

where m_i^T is the i-*th* row of M

Camera calibration: Direct Linear Transform (DLT)

$$\Rightarrow \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} m_1^T \\ m_2^T \\ m_3^T \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \rightarrow P$$

Conversion back from homogeneous coordinates to pixel coordinates leads to:

$$u = \frac{\tilde{u}}{\tilde{w}} = \frac{m_1^T \cdot P}{m_3^T \cdot P}$$
$$v = \frac{\tilde{v}}{\tilde{w}} = \frac{m_2^T \cdot P}{m_3^T \cdot P}$$

$$\Rightarrow \quad (m_1^T - u_i m_3^T) \cdot P_i = 0$$
$$(m_2^T - v_i m_3^T) \cdot P_i = 0$$

Camera calibration: Direct Linear Transform (DLT)

By re-arranging the terms, we obtain

$$\begin{aligned}(m_1^T - u_i m_3^T)\cdot P_i = 0 \\ (m_2^T - v_i m_3^T)\cdot P_i = 0\end{aligned} \Rightarrow \begin{pmatrix} P_1^T & 0^T & -u_1 P_1^T \\ 0^T & P_1^T & -v_1 P_1^T \end{pmatrix}\begin{pmatrix} m_1 \\ m_2 \\ m_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

For n points, we can stack all these equations into a big matrix:

$$\begin{pmatrix} P_1^T & 0^T & -u_1 P_1^T \\ 0^T & P_1^T & -v_1 P_1^T \\ \cdots & \cdots & \cdots \\ P_n^T & 0^T & -u_n P_n^T \\ 0^T & P_n^T & -v_n P_n^T \end{pmatrix}\begin{pmatrix} m_1 \\ m_2 \\ m_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}$$

Camera calibration: Direct Linear Transform (DLT)

By re-arranging the terms, we obtain

$$\begin{aligned}(m_1^T - u_i m_3^T)\cdot P_i = 0 \\ (m_2^T - v_i m_3^T)\cdot P_i = 0\end{aligned} \Rightarrow \begin{pmatrix} P_1^T & 0^T & -u_1 P_1^T \\ 0^T & P_1^T & -v_1 P_1^T \end{pmatrix}\begin{pmatrix} m_1 \\ m_2 \\ m_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

For n points, we can stack all these equations into a big matrix:

$$\begin{pmatrix} X_w^1 & Y_w^1 & Z_w^1 & 1 & 0 & 0 & 0 & 0 & -u_1 X_w^1 & -u_1 Y_w^1 & -u_1 Z_w^1 & -u_1 \\ 0 & 0 & 0 & 0 & X_w^1 & Y_w^1 & Z_w^1 & 1 & -v_1 X_w^1 & -v_1 Y_w^1 & -v_1 Z_w^1 & -v_1 \\ & & & & & & \cdots & & \cdots & \cdots & \cdots & \\ X_w^n & Y_w^n & Z_w^n & 1 & 0 & 0 & 0 & 0 & -u_n X_w^n & -u_n Y_w^n & -u_n Z_w^n & -u_n \\ 0 & 0 & 0 & 0 & X_w^n & Y_w^n & Z_w^n & 1 & -v_n X_w^n & -v_n Y_w^n & -v_n Z_w^n & -v_n \end{pmatrix} \begin{pmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{14} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{24} \\ m_{31} \\ m_{32} \\ m_{33} \\ m_{34} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} \Rightarrow \mathrm{Q}\cdot\mathrm{M} = 0$$

Q (this matrix is **known**)

M (this matrix is **unknown**)

Camera calibration: Direct Linear Transform (DLT)

$$\mathrm{Q} \cdot \mathrm{M} = 0$$

Minimal solution

- Q has 11 Degrees of Freedom (in fact, Q is valid up to a scale factor, thus, 12-1 = 11)
- Each 3D-to-2D point correspondence provides 2 independent equations
- Thus, $5+\frac{1}{2}$ point correspondences are needed (in practice **6 point** correspondences!)

Over-determined solution

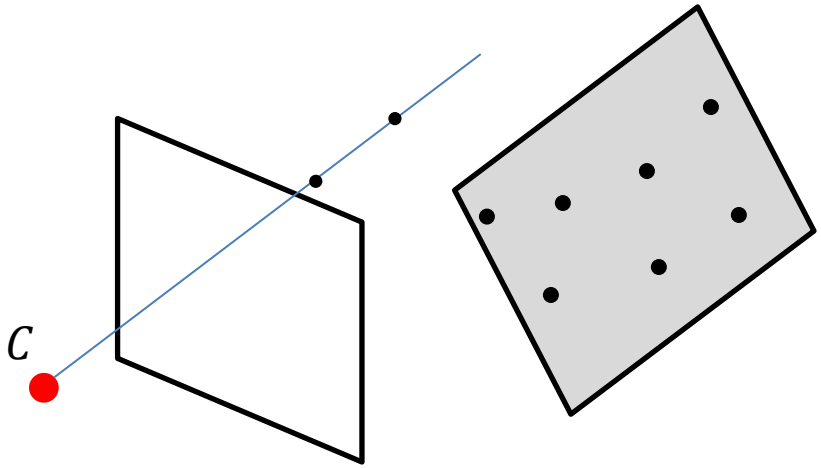
- $n \geq 6$ points
- A solution is to minimize $||QM||$ subject to the constraint $||M||^2 = 1$.
  It can be solved through Singular Value Decomposition (SVD). The solution is the eigenvector corresponding to the smallest eigenvalue of the matrix $Q^T Q$ (because it is the unit vector x that minimizes $x^T Q^T Q x$).

Camera calibration: Direct Linear Transform (DLT)

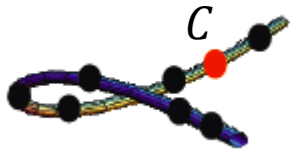
$$\mathrm{Q} \cdot \mathrm{M} = 0$$

Degenerate configurations

1. Points lying on a **plane** and/or along a single **line** passing through the **projection center**



2. Camera and points on a twisted cubic (i.e., smooth curve in 3D space of degree 3)



Camera calibration: Direct Linear Transform (DLT)

- Once we have the M matrix, we can recover the intrinsic and extrinsic parameters by remembering that

$$\mathrm{M} = \mathrm{K}(\mathrm{R} \mid \mathrm{T})$$

$$\begin{bmatrix} m_{11} & m_{12} & m_{13} & m_1 \\ m_{21} & m_{22} & m_{23} & m_2 \\ m_{31} & m_{32} & m_{33} & m_3 \end{bmatrix} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}$$

Camera calibration: Direct Linear Transform (DLT)

- Once we have the M matrix, we can recover the intrinsic and extrinsic parameters by remembering that

$$\mathrm{M} = \mathrm{K}(\mathrm{R} \mid \mathrm{T})$$

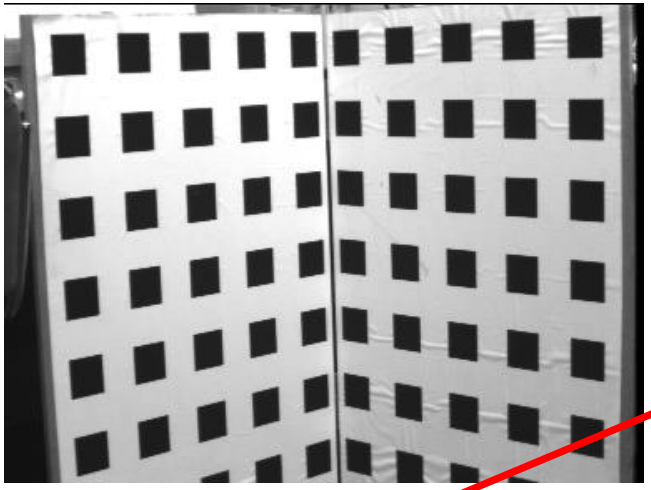
$$\begin{bmatrix} m_{11} & m_{12} & m_{13} & m_1 \\ m_{21} & m_{22} & m_{23} & m_2 \\ m_{31} & m_{32} & m_{33} & m_3 \end{bmatrix} = \begin{bmatrix} \alpha r_{11} + u_0 r_{31} & \alpha r_{12} + u_0 r_{32} & \alpha r_{13} + u_0 r_{33} & \alpha t_1 + u_0 t_3 \\ \alpha r_{21} + v_0 r_{31} & \alpha r_{22} + v_0 r_{32} & \alpha r_{23} + v_0 r_{33} & \alpha t_2 + v_0 t_3 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}$$

- However, notice that we are not enforcing the constraint that R is orthonormal, i.e., $R \cdot R^T = I$

- To do this, we can use the so-called QR factorization of M, which decomposes M into a R (orthonormal), T, and an upper triangular matrix (i.e., K)

Tsai's (1987) Calibration example

1. Edge detection
2. Straight line fitting to the detected edges
3. Intersecting the lines to obtain the images corners (corner accuracy <0.1 pixels!)
4. Use >6 points



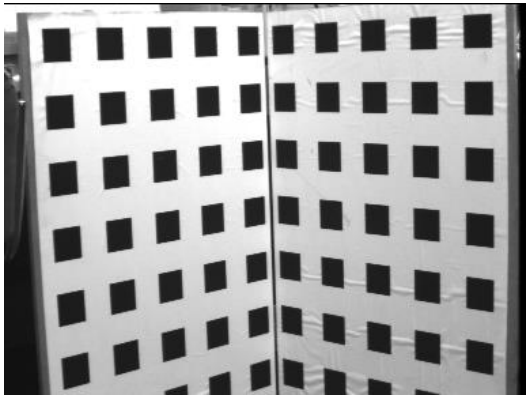
Why is this ratio not 1?

What are the «skew» and «residuals»?

f_y	f_x/f_y	skew	x_0	y_0	residual
1673.3	1.0063	1.39	379.96	305.78	0.365

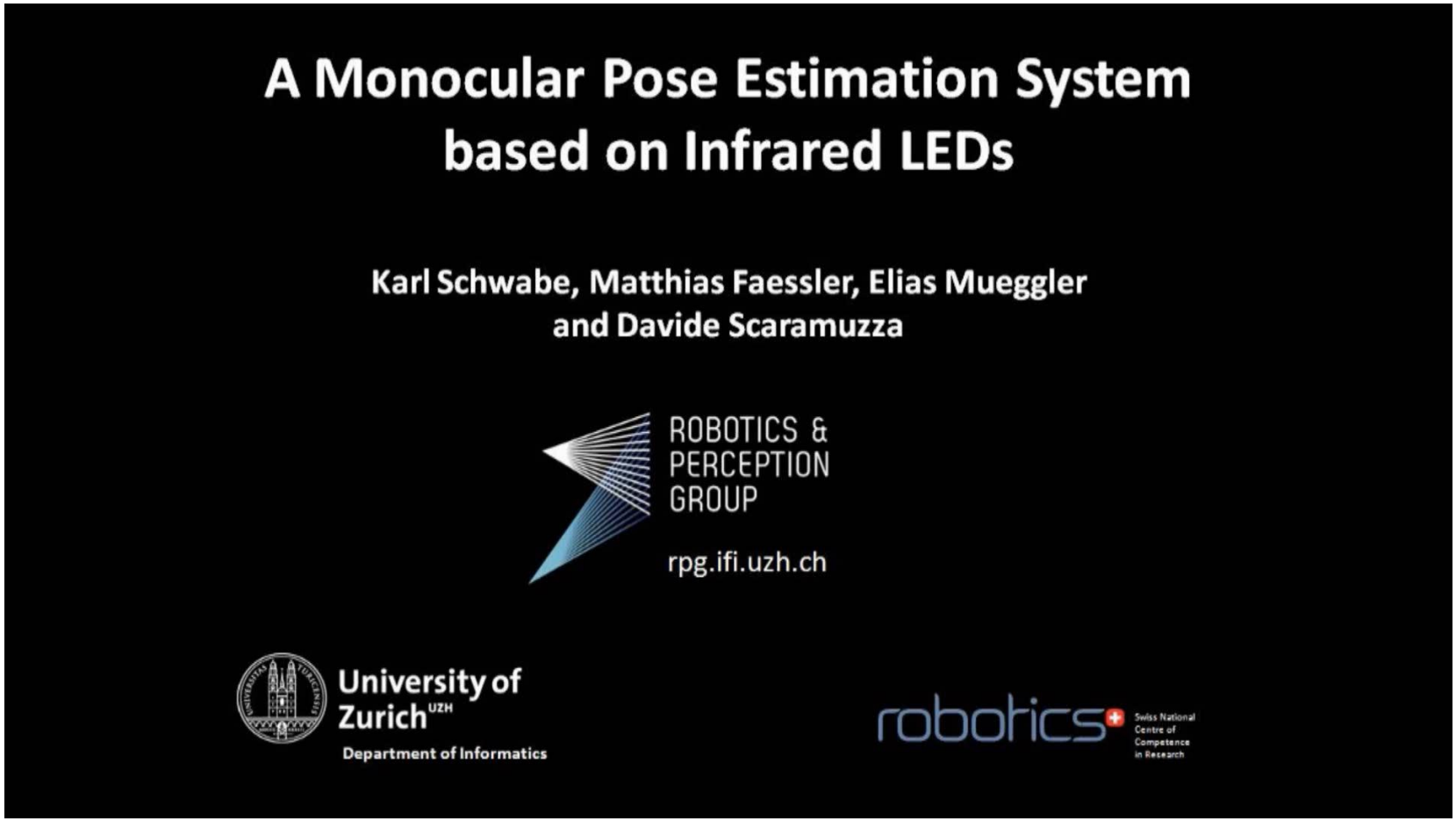
Tsai's (1987) Calibration example

- The original Tsai calibration (1987) used to consider two different focal lengths α_u, α_v (which means that the pixels are not squared) and a skew factor ($K_{12} \neq 0$, which means the pixes are parallelograms instead of rectangles). This relaxation was used to account for possible misalignments between CCD and lens
- Most of today's camera are well manufactured, thus, we can assume $\frac{\alpha_u}{\alpha_v} = 1$ and $K_{12} = 0$
- What is the residual? The residual is the *average* "reprojection error". The reprojection error is computed as the distance (in pixels) between the observed pixel point and the camera-reprojected 3D point. The reprojection error gives as a quantitative measure of the accuracy of the calibration (ideally it should be zero).



f_y	f_x/f_y	skew	x_0	y_0	residual
1673.3	1.0063	1.39	379.96	305.78	0.365

DLT algorithm applied to mutual robot localization



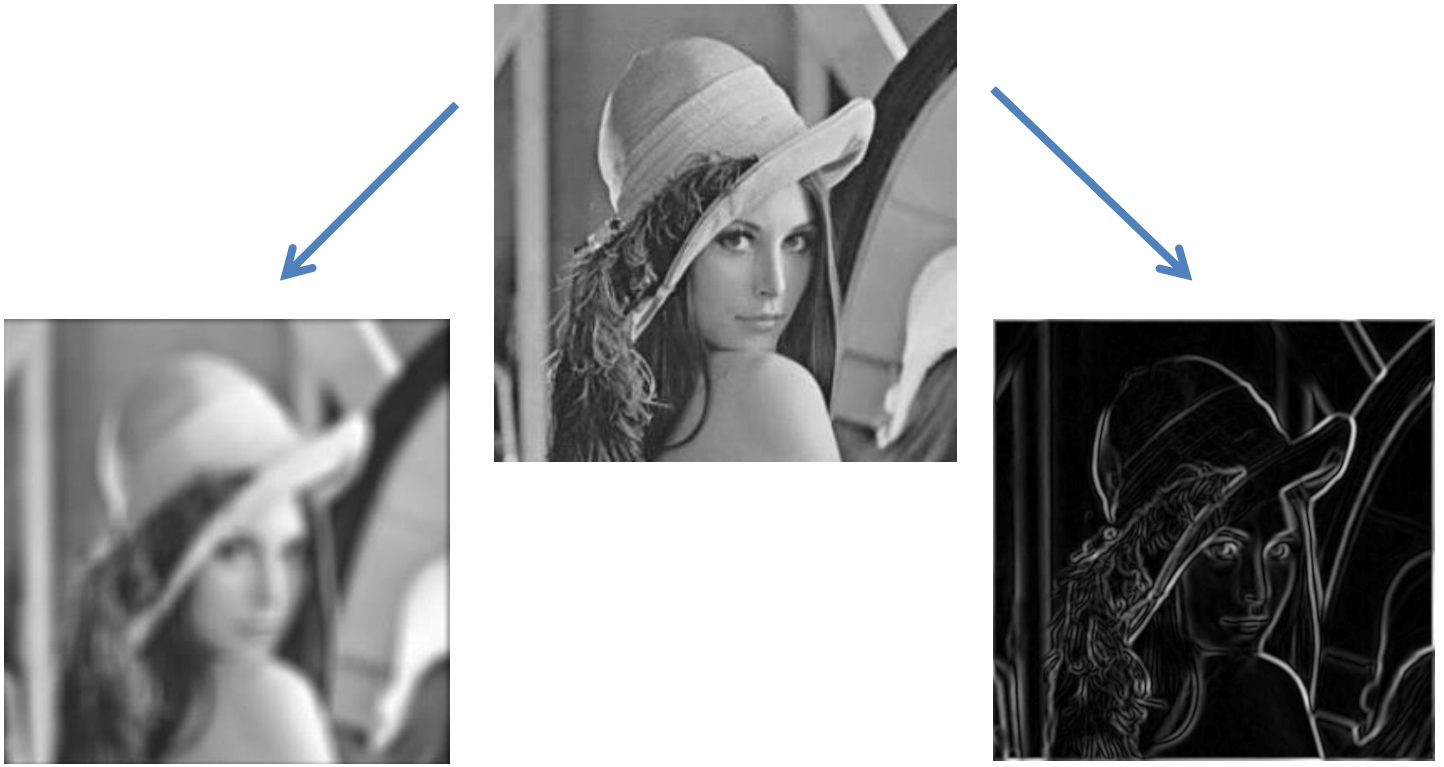
In this case, the camera has been pre-calibrated (i.e., K is known). Can you think of how the DLT algorithm could be modified so that only R and T need to determined and not K?

Summary (things to remember)

- Perspective Projection Equation
- Intrinsic and extrinsic parameters (K, R, t)
- Homogeneous coordinates
- Normalized image coordinates
- Image formation equations (including radial distortion)
- Camera calibration
  - DLT algorithm (for **non planar** scenes)
- Readings for today:
  - Chapter 2.1 of Szeliski book (freely downloadable from http://szeliski.org/Book/
  - Chapters 4.1-4.3 of Autonomous Mobile Robots book

Next time

- Camera calibration
  - from planar grids
- Non conventional camera models
- Filtering and Edge Detection
- Readings for next lecture: 3.2, page 108-109, 4.2.1



Low-pass filtered image

High-pass filtered image

Mini Project List

- It is not mandatory; you can either choose a mini project or to present a paper
- We propose a list of projects but you can also propose your own idea
- The project can be done in Matlab, Open CV (C/C++ or Python)

- Please contact Zichao Zhang <zhangzichao17 at gmail dot com> and Elias Mueggler <mueggler at ifi dot uzh dot ch> if you have questions about the proposed projects or want to propose an own idea.

Mini Project List:

http://rpg.ifi.uzh.ch/docs/teaching/Mini_Projects.pdf

- **Count fruits on a tree -** In agriculture monitoring and automated agriculture, detecting and counting fruits (e.g., oranges) is an important building block. The goal is to detect and count a specific fruit on a tree that could be recognized by color, shape, etc.
- **Detect and identify playing cards -** In this project, you detect and identify playing cards from images. This involves detecting basic shapes and template matching. The output of this program could serve as input to a robot that plays poker.
- **Read barcodes -** The goal of this project is to find barcodes in images (e.g., on products) and identify the number they represent. This involves image filter, detection of specific shapes (bars), and interpretation of these bars.
- **Stitch panorama images -** The goal of this project is to create panoramic images from a set of overlapping photographs. This involves finding correspondences between the images and warping them accordingly.
- **Identify the state of a game -** In this project, the goal is to identify the state of a game (e.g., Rubik's Cube, Nine Men's Morris (German: Mühle), or Four Wins). This involves detecting the play field and its elements by shape, color, etc. Such a program could provide the input to a robot that plays games with humans.
- **Estimate the height of a building -** In this project, you are required to estimate the height of the building by counting the number of stories. The height of each story is assumed to be known. This involves image filtering and interpretation.
- **Optical Character Recognition (OCR) -** OCR is an essential module for digitalizing documents and office automation. The goal of this project is to identify individual characters in an image. This involves image filtering, segmentation and template matching.
- **Visual Odometry (VO) -** VO is the process of estimating a camera's motion from the images only. The goal of this project is to implement a visual odometry pipeline. This involves finding image correspondences and motion estimation based on two-view geometry.