TSBB21, Lecture 4b Camera calibration 1

- Camera calibration 1
  - Homogenous matrices for scaling, translation, rotation, skewing
  - The Pinhole camera model
  - Outer and inner parameters
  - 3D calibration of a camera
  - Calibration of a flat world, a homography
  - Inhomogeneous and homogeneous solutions.
  - Camera resectioning

- Literature
  - ”Short about camera geometry and camera calibration” by Maria Magnusson

- Alternative Literature
  - Parts of …
    ”Introduction to Representations and Estimation in Geometry” (IREG) by Klas Nordberg

Transformation with homogenous matrices

- A point in the 3D-world can be described in homogenous coordinates as $(X,Y,Z,1)^T$. It can be transformed to a new point $(X_1,Y_1,Z_1,1)^T$ by using the 4x4-matrix **M** according to:

$$\begin{pmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{pmatrix} = \mathbf{M} \cdot \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

A homogeneous matrix for translation

Translation

$$\mathbf{T}(t_x, t_y, t_z) = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$Eq.\ (5)$

Example:

$$\begin{pmatrix} X + t_x \\ Y + t_y \\ Z + t_z \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Note:
A normal 3x3-matrix will not work for translation!

Homogenous matrices for scaling and skewing

Scaling

$$\mathbf{S}(s_a, s_b, s_c) = \begin{pmatrix} s_a & 0 & 0 & 0 \\ 0 & s_b & 0 & 0 \\ 0 & 0 & s_c & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Eq. (3)

Skewing in the x-direction depending on the y-coordinate

$$\begin{pmatrix} 1 & a & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Eq. (10)

General skewing

$$\begin{pmatrix} 1 & a & b & 0 \\ c & 1 & d & 0 \\ e & f & 1 & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Eq. (11)

Homogeneous matrices for rotation

Rotation with the angle θ around the x-axis

Eq. (7)

$$\mathbf{R}_x = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Eq. (8)

Rotation with the angle θ around the y-axis

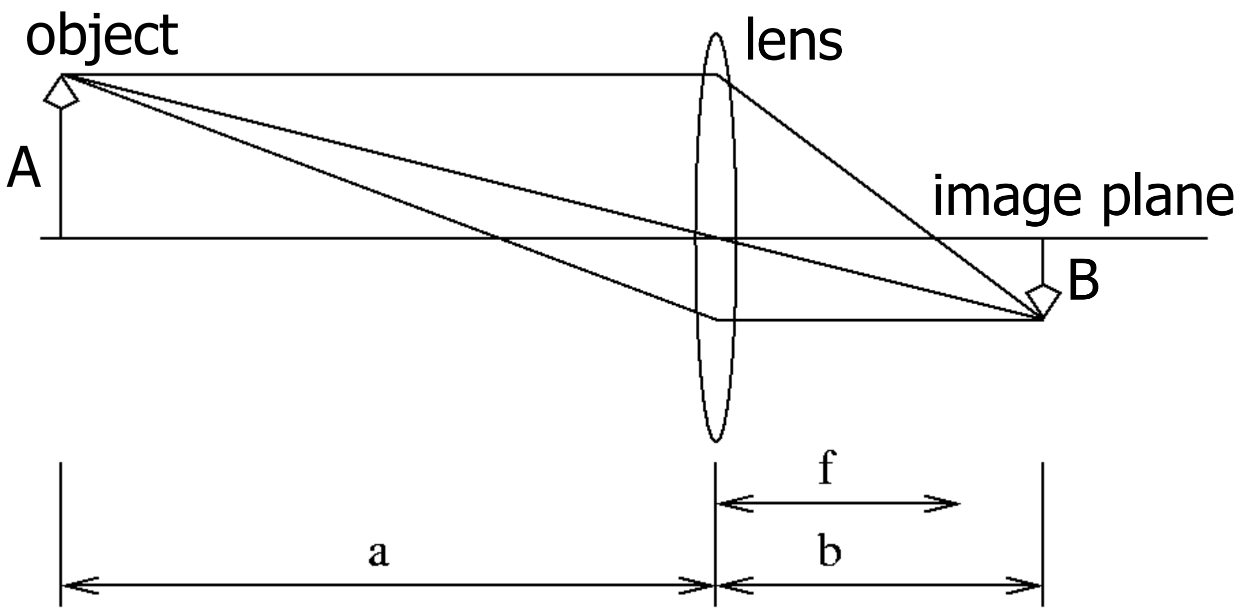
$$\mathbf{R}_y = \begin{pmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotation with the angle θ around the z-axis

Eq. (9)

$$\mathbf{R}_z = \begin{pmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The Lens law (repetition)



The lens law:

$$\frac{1}{a}+\frac{1}{b}=\frac{1}{f}$$

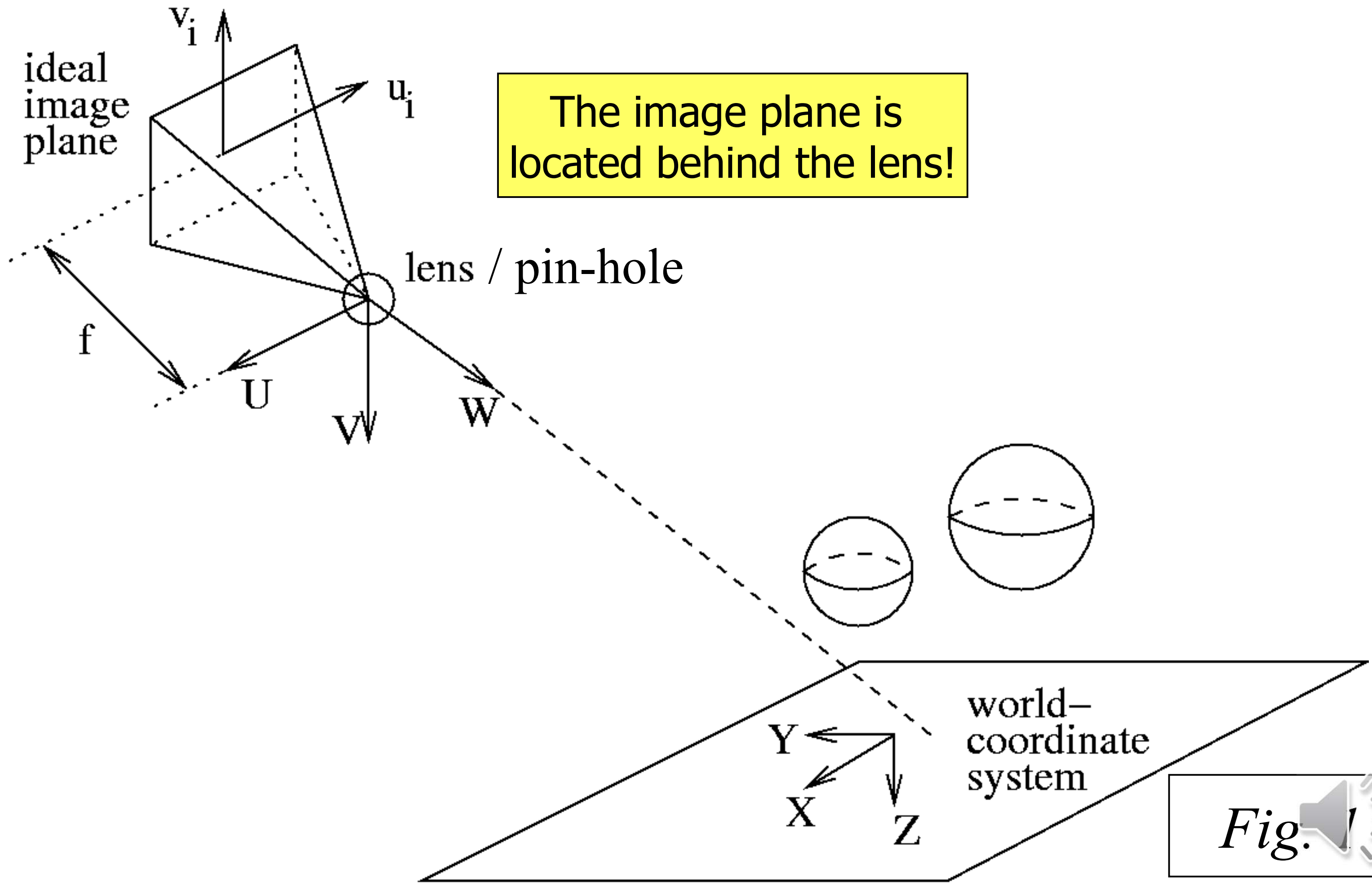
where f is the focal length

Size relations:

$$\frac{A}{a}=\frac{B}{b}\approx\frac{B}{f}$$

The lens law states that if the image plane is located at the distance b from the lens, then the object at distance a from the lens will give a sharp image.
Note that since normally a>>b => b≈f.

The pinhole camera model, real geometry



The image plane is located behind the lens!

Fig. 1

The pinhole camera model, mirrored

Here we use the notation: **ideal image plane** with coordinates (u_i, v_i). Alternatively the notation **normalized image plane** with coordinates $(u_n, v_n) = (u_i/f, v_i/f)$ may be used.

The image plane is mirrored so that it is located in front of the lens.

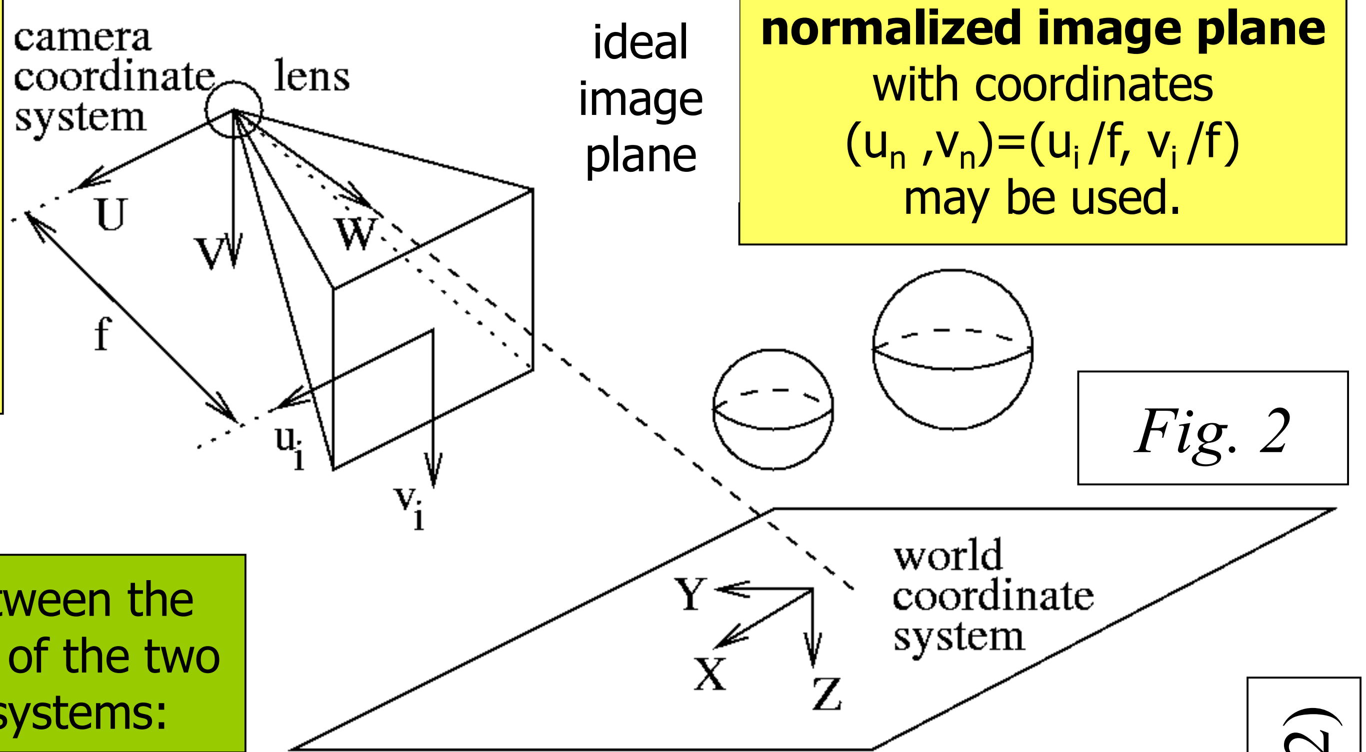


Fig. 2

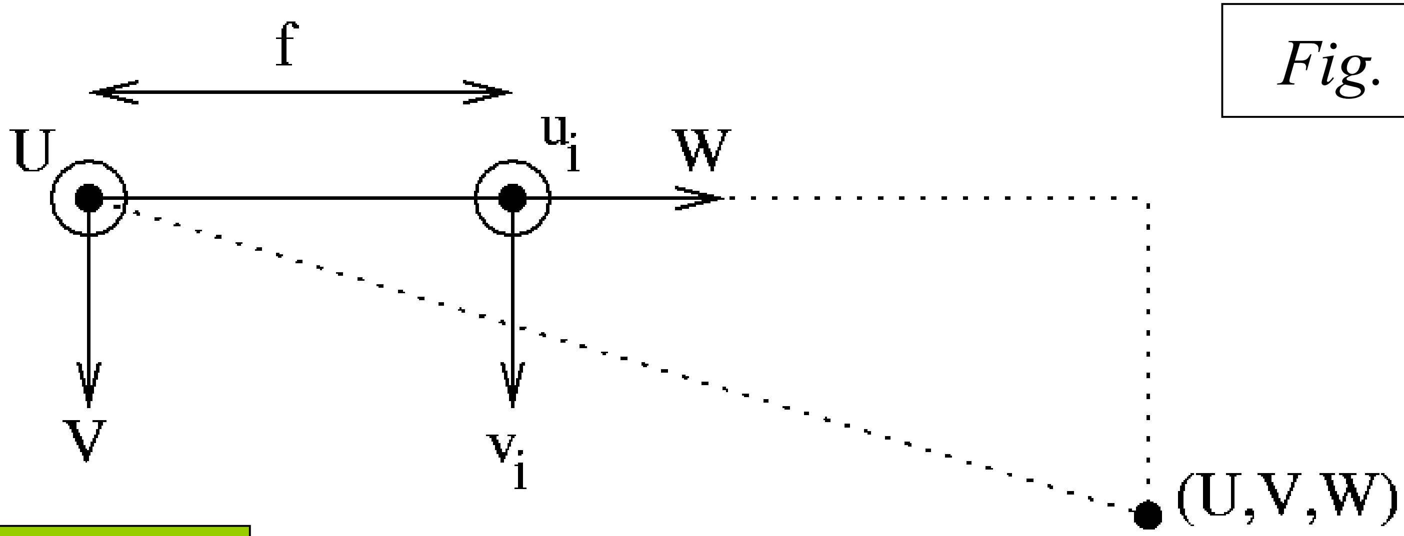
Relation between the coordinates of the two coordinate systems:

$$W(u_n, v_n, 1)^T = W\left(\frac{u_i}{f}, \frac{v_i}{f}, 1\right)^T = (U, V, W)^T = [\mathbf{Rt}] \cdot (X, Y, Z, 1)^T$$

Eq. (12)

Technique to express perspective transformation with vectors

Fig. 3



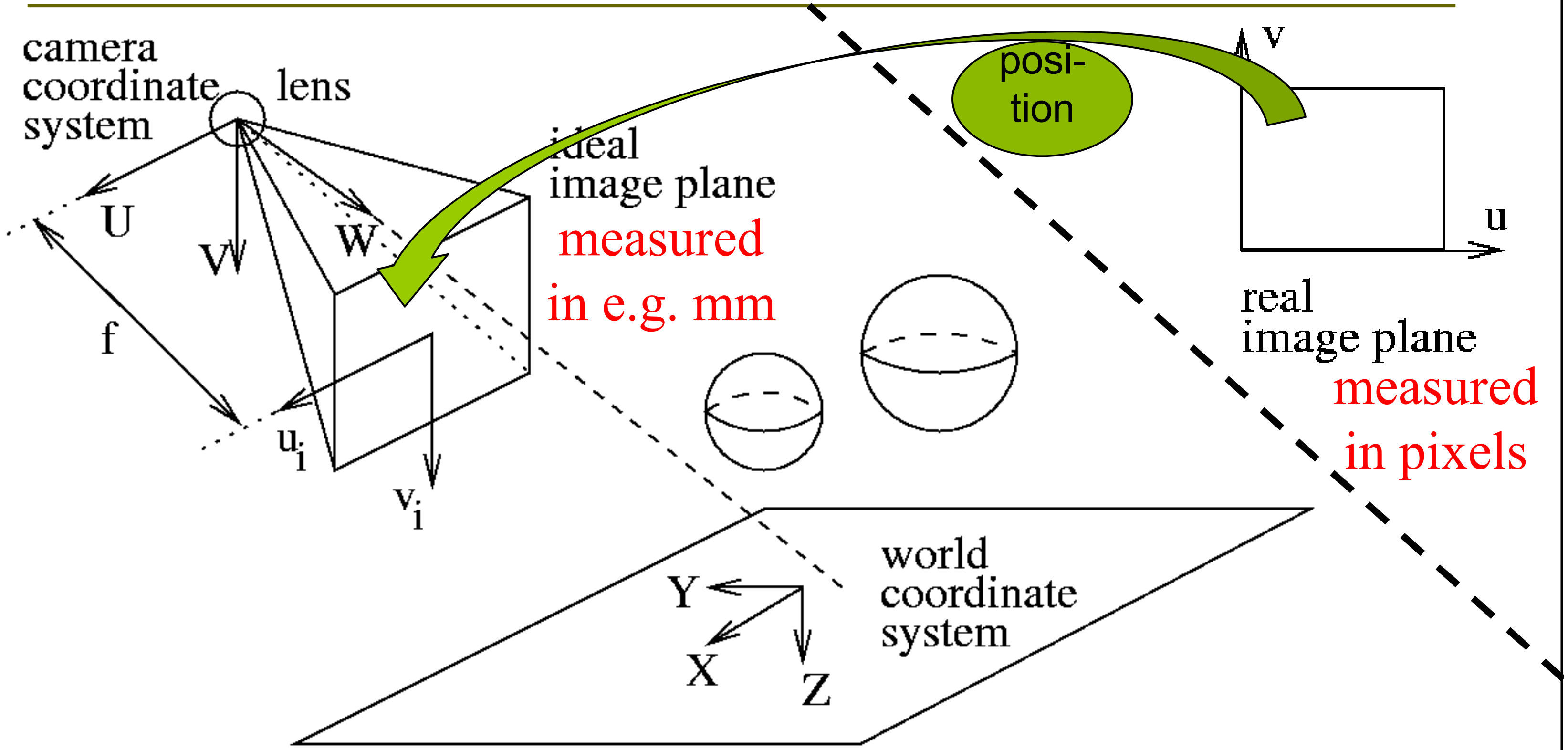
Uniform triangles gives:

u_n and v_n are the norma-lized image coordinates

$$\begin{cases} v_n = \dfrac{v_i}{f} = \dfrac{V}{W} \\ u_n = \dfrac{u_i}{f} = \dfrac{U}{W} \end{cases} \Rightarrow \quad W\left(\frac{u_i}{f}, \frac{v_i}{f}, 1\right)^T = (U, V, W)^T$$

Eq. (14)

Relation between the ideal image plane and the real image plane



Eq. (15)

$$(u, v, 1)^T = \mathbf{A} \cdot \left(\frac{u_i}{f}, \frac{v_i}{f}, 1\right)^T$$

Relation between world coordinates and real image coordinates

Relation between the coordinate systems:

$$W\left(\frac{u_i}{f},\frac{v_i}{f},1\right)^T = (U,V,W)^T$$
$$= [\mathbf{Rt}]\cdot(X,Y,Z,1)^T$$

Eq. (12)

Relation between the image planes:

$$(u,v,1)^T = \mathbf{A}\cdot\left(\frac{u_i}{f},\frac{v_i}{f},1\right)^T$$

Eq. (15)

Relation between world coordinates and real image coordinates:

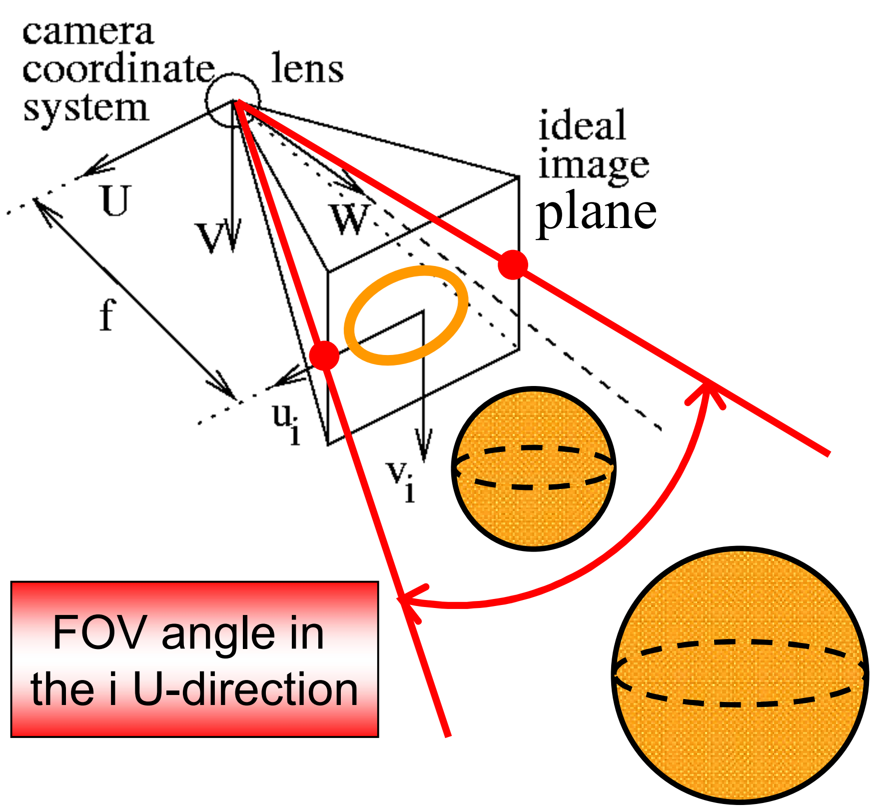
$$(u,v,1)^T \sim s(u,v,1)^T = \mathbf{A}[\mathbf{Rt}]\cdot(X,Y,Z,1)^T$$

equivalence

Eq. (18)

Note that s replaces W as "perspective projection parameter"

Unambiguousness, field of view (FOV) and resolution



- The parameters W and s are not unambiguously determined. Both the big ball and the small ball gives the same contour in the (u_i,v_i)-plane. Consequently, we cannot know W.
- Therefore we can also change W to s in the previous slide.
- It is appropriate to measure the field of view (FOV) as the largest measurable angle in the U- and V-direction. (see e.g. Lab exercise E: Panorama stitching)
- The resolution of an object in an image depends on the distance from the camera. The resolution in the U-direction can, for example, be measured as the FOV angle/the number of pixels.

Inner and outer parameters

Relation between world coordinates and real image coordinates:

$$(u, v, 1)^T \sim s(u, v, 1)^T = \mathbf{A}[\mathbf{Rt}] \cdot (X, Y, Z, 1)^T$$

Inner parameters

Outer parameters

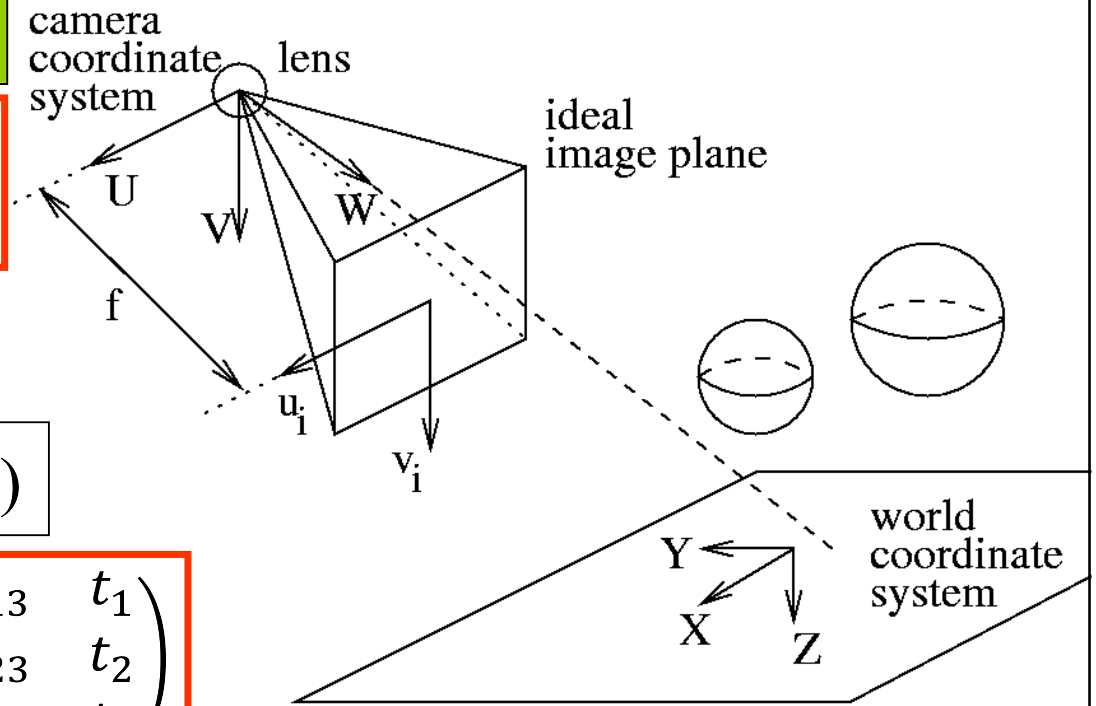
The inner parameters for a camera can be determined through a calibration procedure.

The outer parameters for a camera **at a fix position** can be determined through a calibration procedure.

Outer parameters

Relation between the coordinate systems:

$$(U,V,W)^T = [\mathbf{Rt}] \cdot (X,Y,Z,1)^T$$



Eq. (12) *Eq.* (13)

$$[\mathbf{Rt}] = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{pmatrix}$$

(t_1,t_2,t_3): the translation of the camera in relation to the world
R: the rotation of the camera in relation to the world

Inner parameters

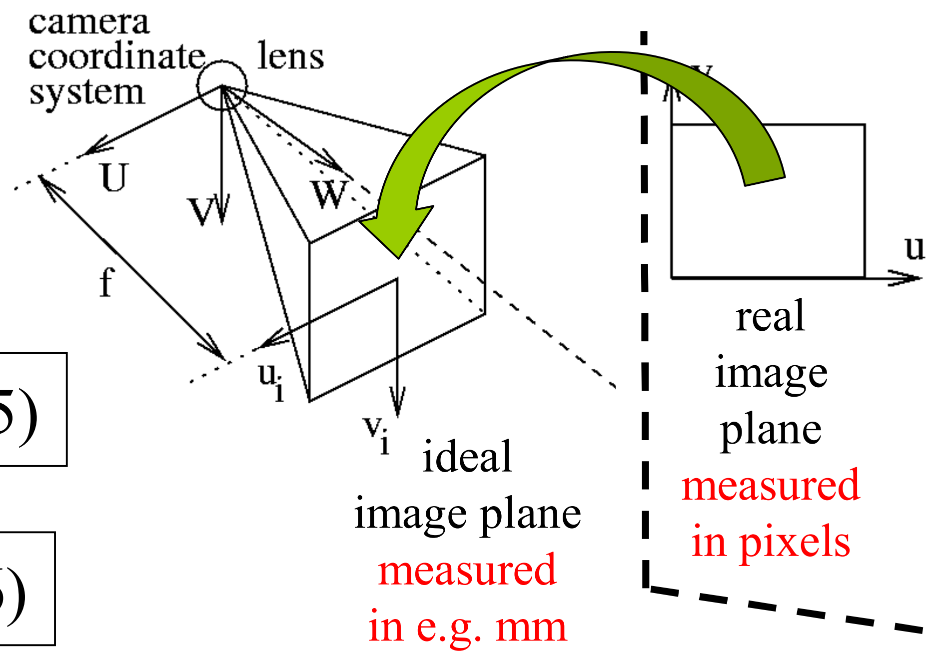
Relation between the image planes:

$$(u, v, 1)^T = \mathbf{A} \cdot \left(\frac{u_i}{f}, \frac{v_i}{f}, 1 \right)^T$$

Eq. (15)

$$\mathbf{A} = \begin{pmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{pmatrix}$$

Eq. (16)

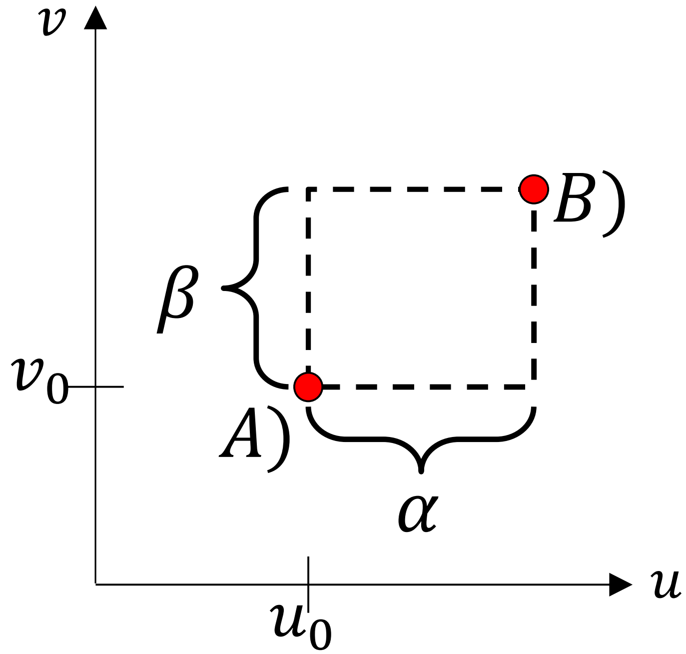


β: scaling in the v-direction
α: scaling i the u-direction
γ: skewing (lack of orthogonality between horizontal and vertical axes) (often close to 0)
(u_0, v_0): the cross-section between the optical axis and the real image plane

Inner parameters, ex) with γ=0

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u_i/f \\ v_i/f \\ 1 \end{pmatrix}$$

	(u,v)	(u_i,v_i)
$A)$	(u_0,v_0)	$(0,0)$
$B)$	$(\alpha+u_0,\beta+v_0)$	(f,f)

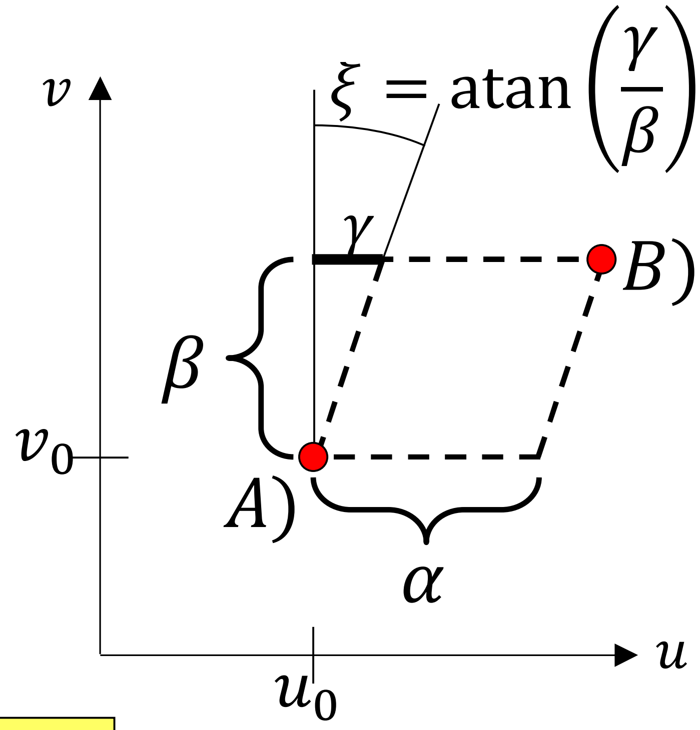


(u_0,v_0): the cross-section between the optical axis and the real image plane, the image center, the principal point.
α and β denotes the scaling i the u- and v-direction, respectively.
If $\alpha = \beta$, the pixels are quadratic.
If $\alpha \neq \beta$, the pixels are rectangular, but not quadratic.

Inner parameters, ex) with $\gamma \neq 0$

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u_i/f \\ v_i/f \\ 1 \end{pmatrix}$$

	(u, v)	(u_i, v_i)
$A)$	(u_0, v_0)	$(0,0)$
$B)$	$(\alpha + \gamma + u_0, \beta + v_0)$	(f, f)



γ is the skewing parameter
ξ =arctan(γ/β) gives an angular measurement
ξ is normally small, i.e. close to 0 degrees

3D calibration of a camera

$$s(u, v, 1)^T = \mathbf{A}[\mathbf{Rt}] \cdot (X, Y, Z, 1)^T \qquad Eq. (17)$$

$$s(u, v, 1)^T = \mathbf{C} \cdot (X, Y, Z, 1)^T \qquad Eq. (18)$$

We will first determine **C**, only.
Later, we will learn how to determine **A**, **R** and **t**.

Depending on the variable s, **C** can only be determined up to a scale factor, say λ.

$$\mathbf{C} = \begin{pmatrix} C_{11} & C_{12} & C_{13} & C_{14} \\ C_{21} & C_{22} & C_{23} & C_{24} \\ C_{31} & C_{32} & C_{33} & C_{34} \end{pmatrix}$$

We have now two possibilites,
either make an **inhomogeneous** or an **homogeneous** solution.

3D calibration, the inhomogeneous solution

- Set C_{34} = 1. (If C_{34} seems to be 0, another element can be set to 1.)

$$\mathbf{C} = \begin{pmatrix} C_{11} & C_{12} & C_{13} & C_{14} \\ C_{21} & C_{22} & C_{23} & C_{24} \\ C_{31} & C_{32} & C_{33} & \mathbf{1} \end{pmatrix} \qquad \boxed{Eq.\ (20)}$$

The matrix **C** can be determined by measuring a number of corresponding point (how many?) in the world (X_i, Y_i, Z_i) and the image (u_i, v_i), where $1 \leq i \leq N$.

Inhomogeneous solution

$$s(u,v,1)^T = \mathbf{C}\cdot(X,Y,Z,1)^T$$

$$\mathbf{C} = \begin{pmatrix} C_{11} & C_{12} & C_{13} & C_{14} \\ C_{21} & C_{22} & C_{23} & C_{24} \\ C_{31} & C_{32} & C_{33} & \mathbf{1} \end{pmatrix}$$

Set:

$Eq.\ (21)$

$Eq.\ (20)$

$Eq.\ (19)$

$$\mathbf{c} = (C_{11}, C_{12}, C_{13}, C_{14}, C_{21}, C_{22}, C_{23}, C_{24}, C_{31}, C_{32}, C_{33})$$

$$\mathbf{D}\cdot\mathbf{c} =$$

$$\begin{pmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -u_1X_1 & -u_1Y_1 & -u_1Z_1 \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -v_1X_1 & -v_1Y_1 & -v_1Z_1 \\ X_2 & Y_2 & Z_2 & 1 & 0 & 0 & 0 & 0 & -u_2X_2 & -u_2Y_2 & -u_2Z_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & X_N & Y_N & Z_N & 1 & -v_NX_N & -v_NY_N & -v_NZ_N \end{pmatrix} \begin{pmatrix} C_{11} \\ C_{12} \\ C_{13} \\ \vdots \\ C_{33} \end{pmatrix} = \begin{pmatrix} u_1 \\ v_1 \\ u_2 \\ \vdots \\ v_N \end{pmatrix}$$

$$= \mathbf{f}$$

$Eq.\ (22)$

11 equations give that at least 6 point-pairs ("5½") is needed to determine **C**

Show Eq. (21)

Show Eq (21)

We have measured the point (X_1, Y_1, Z_1) in the world.

It corresponds to (u_1, v_1) in the image.

$$(18, 19) \Rightarrow \quad s\begin{pmatrix} u_1 \\ v_1 \\ 1 \end{pmatrix} = \begin{pmatrix} C_{11} & C_{12} & C_{13} & C_{14} \\ C_{21} & C_{22} & C_{23} & C_{24} \\ C_{31} & C_{32} & C_{33} & 1 \end{pmatrix} \begin{pmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{pmatrix}$$

$$s u_1 = C_{11} X_1 + C_{12} Y_1 + C_{13} Z_1 + C_{14} \quad (a)$$

$$s v_1 = \cdots \quad (b)$$

$$s = C_{31} X_1 + C_{32} Y_1 + C_{33} Z_1 + 1 \quad (c)$$

$$(a, c) \Rightarrow \quad u_1 = C_{11} X_1 + C_{12} Y_1 + C_{13} Z_1 + C_{14} - C_{31} X_1 u_1 - C_{32} Y_1 u_1 - C_{33} Z_1 u_1$$

the first row in (21)

Solution of the equation system

If we measure 5½ point-pairs, we get 11 equations.
The equation system can be solved as:

$$\boxed{\mathbf{c} = \mathbf{D}^{-1} \cdot \mathbf{f}}$$

If we measure more than 5½ point-pairs, the equation system becomes over-determined with the solution:

More point-pairs gives a more certain solution!

D⁺ is pinv in Matlab

$$\boxed{\begin{array}{ccc} \mathbf{D} \cdot \mathbf{c} & = & \mathbf{f} \\ \mathbf{D}^T\mathbf{D} \cdot \mathbf{c} & = & \mathbf{D}^T\mathbf{f} \\ \mathbf{c} & = & (\mathbf{D}^T\mathbf{D})^{-1}\mathbf{D}^T\mathbf{f} \\ \mathbf{c} & = & \mathbf{D}^+\mathbf{f} \end{array}} \qquad Eq.\ (23)$$

D$^+$ is the so called pseudo-inverse of **D**.
This is the Least Square solution of the equation system.
This is also equivalent to Maximum Likelihood-minimization.

3D calibration, the homogeneous solution

- In the homogeneous solution, C_{34} is not set to 1. Instead **C** is kept as:

$$\mathbf{C} = \begin{pmatrix} C_{11} & C_{12} & C_{13} & C_{14} \\ C_{21} & C_{22} & C_{23} & C_{24} \\ C_{31} & C_{32} & C_{33} & \color{red}{C_{34}} \end{pmatrix}$$

- To improve performance, Hartley normalization (see e.g. IREG) is used:
  - (X_i, Y_i, Z_i)-coordinates:
    - Calculate the mean and standard deviation.
    - Subtract the mean, divide by the standard deviation and multiply with $\sqrt{2}$
  - (u_i, v_i)-coordinates:
    - Calculate the mean and standard deviation.
    - Subtract the mean, divide by standard dev. and mult. with $\sqrt{2}$
- Form an equation system, see next slide.
- Solve using SVD, see next-next lecture.

Homogeneous solution

$$s(u,v,1)^T = \mathbf{C}\cdot(X,Y,Z,1)^T$$

$$\mathbf{C} = \begin{pmatrix} C_{11} & C_{12} & C_{13} & C_{14} \\ C_{21} & C_{22} & C_{23} & C_{24} \\ C_{31} & C_{32} & C_{33} & C_{34} \end{pmatrix}$$

$Eq.\ (19)$

Set:

$$\mathbf{c} = (C_{11}, C_{12}, C_{13}, C_{14}, C_{21}, C_{22}, C_{23}, C_{24}, C_{31}, C_{32}, C_{33}, C_{34})$$

$$\mathbf{D}\cdot\mathbf{c} =$$

$$\begin{pmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -u_1X_1 & -u_1Y_1 & -u_1Z_1 & -u_1 \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -v_1X_1 & -v_1Y_1 & -v_1Z_1 & -v_1 \\ X_2 & Y_2 & Z_2 & 1 & 0 & 0 & 0 & 0 & -u_2X_2 & -u_2Y_2 & -u_2Z_2 & -u_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & X_N & Y_N & Z_N & 1 & -v_NX_N & -v_NY_N & -v_NZ_N & -v_N \end{pmatrix} \begin{pmatrix} C_{11} \\ C_{12} \\ C_{13} \\ \vdots \\ C_{34} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Matlab solution:

```
[U,S,V] = svd(D);
c = V(:,12);
```

c may then be scaled, if desired

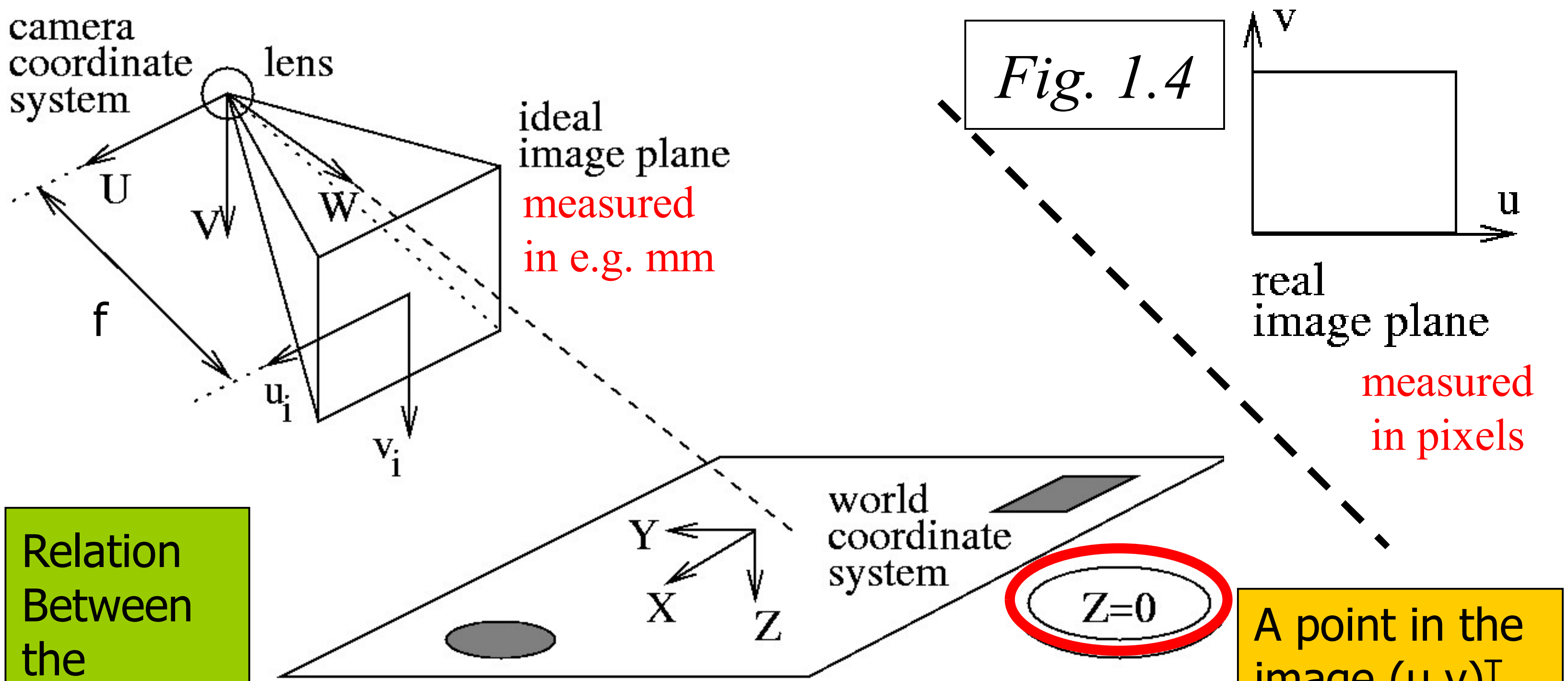
From C to A[Rt]

- When the matrix C is determined, it is possible to receive A, R and t by using a little linear algebra.
- This procedure is called camera resectioning.
- We will talk about that in the end of this lecture.

Using the calibrated camera

- We now know how a point in the world $(X,Y,Z)^T$ will be mapped to a point in the image $(u,v)^T$.
- We do **not** know how a point in the image $(u,v)^T$ will be mapped to a point in the world $(X,Y,Z)^T$.
- But we do know that a point in the image $(u,v)^T$ corresponds to a **line** in the world $(X,Y,Z)^T$.
- From A and an object point in the image, we can calculate the **angular direction** to the corresponding object point in the world. Then it is possible for a movable camera to follow an object. Lab task!
- If we have more knowledge about the world, for example if it is a **flat** world, we know that a point in the image $(u,v)^T$ is mapped to a point in the world $(X,Y,Z)^T$. This is camera calibration of a flat world, a **homography**. Lab task!
- Another possibility is to use **stereo**, i.e. using two calibrated cameras. They gives one straight **line**, each. The cross-section between these lines gives the exact position of the point in the world.

Calibration of a flat world, a homography



Relation Between the coordinate systems:

$$s(u, v, 1)^T = \mathbf{C} \cdot (X, Y, 1)^T$$

Eq. (24)

A point in the image $(u,v)^T$ is mapped to a point in the world $(X,Y,0)^T$ and vice versa.

Inhomogeneous solution of a homography

$$s(u,v,1)^T = \mathbf{C} \cdot (X,Y,1)^T$$

Eq. (24)

$$\mathbf{C} = \begin{pmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & \mathbf{1} \end{pmatrix}$$

Eq. (25)

Set:

$$\mathbf{c} = (C_{11}, C_{12}, C_{13}, C_{21}, C_{22}, C_{23}, C_{31}, C_{32})$$

Eq. (26)

Eq. (27)

$$\mathbf{D} \cdot \mathbf{c} = \begin{pmatrix} X_1 & Y_1 & 1 & 0 & 0 & 0 & -u_1X_1 & -u_1Y_1 \\ 0 & 0 & 0 & X_1 & Y_1 & 1 & -v_1X_1 & -v_1Y_1 \\ X_2 & Y_2 & 1 & 0 & 0 & 0 & -u_2X_2 & -u_2Y_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & X_N & Y_N & 1 & -v_NX_N & -v_NY_N \end{pmatrix} \begin{pmatrix} C_{11} \\ C_{12} \\ C_{13} \\ \vdots \\ C_{32} \end{pmatrix} = \begin{pmatrix} u_1 \\ v_1 \\ u_2 \\ \vdots \\ v_N \end{pmatrix} = \mathbf{f}$$

Solution as before: $\mathbf{c} = \mathbf{D}^+\mathbf{f}$

Matlab solution: `c = pinv(D)*f;`

8 equations give that at least 4 point-pairs is needed to determine **C**

Homogeneous solution of a homography

Note: Hartley normalization (see a previous slide) may improve per-formance!

$$s(u, v, 1)^T = \mathbf{C} \cdot (X, Y, 1)^T$$

$$\mathbf{C} = \begin{pmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{pmatrix}$$

Set:

$$\mathbf{c} = (C_{11}, C_{12}, C_{13}, C_{21}, C_{22}, C_{23}, C_{31}, C_{32}, C_{33})$$

$$\mathbf{D} \cdot \mathbf{c} = \begin{pmatrix} X_1 & Y_1 & 1 & 0 & 0 & 0 & -u_1 X_1 & -u_1 Y_1 & -u_1 \\ 0 & 0 & 0 & X_1 & Y_1 & 1 & -v_1 X_1 & -v_1 Y_1 & -v_1 \\ X_2 & Y_2 & 1 & 0 & 0 & 0 & -u_2 X_2 & -u_2 Y_2 & -u_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & X_N & Y_N & 1 & -v_N X_N & -v_N Y_N & -v_N \end{pmatrix} \begin{pmatrix} C_{11} \\ C_{12} \\ C_{13} \\ \vdots \\ C_{33} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Matlab solution:

```
[U,S,V] = svd(D);
c = V(:,9);
```

c may then be scaled, if desired

Camera resectioning

From previous slides:
Relation between world coordinates and real image coordinates:

$$(u, v, 1)^T \sim \mathbf{A}[\mathbf{Rt}] \cdot (X, Y, Z, 1)^T \sim \mathbf{C} \cdot (X, Y, Z, 1)^T$$

$$\mathbf{A}[\mathbf{Rt}] \sim \mathbf{C}$$

$$\mathbf{C} = \begin{pmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \end{pmatrix}$$

$$\mathbf{A} = \begin{pmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$[\mathbf{Rt}] = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{pmatrix}$$

Camera resectioning

- If **C** is not at infinity then we can always find a unique decomposition of **C** into its internal **A** and external [**Rt**] parameters. This decomposition is referred to as camera resectioning.
- **A** is an **upper triangular** 3x3 matrix
- **R** is a **rotational matrix**, which describes rotations around the X, Y- and Z-axes.
- **R** is also an **orthogonal matrix**, which is a square matrix whose columns and rows are orthogonal unit vectors (i.e. orthonormal vectors), i.e. $\mathbf{R}^T\mathbf{R}=\mathbf{R}\mathbf{R}^T=\mathbf{I}$, where **I** is the identity matrix.
- **t** is a translation vector, which describes a translation along the X, Y- and Z-axes.

QR- and RQ-factorization

- QR-factorization decomposes a matrix **B** into an orthogonal matrix **Q** multiplied by an upper (or right) triangular matrix **R**.
- Matlab command: `[Q,R] = qr(B);`
- **B** and **Q** are m-by-n
- With a trick (see Matlab code later) an **rq** function can be formed, with Matlab command: `[R,Q] = rq(B);`
- In our case:
- `[A,R] = rq(C(:,1:3));`

Confusion:
R has different meanings!
The triangular **R** is marked with turquoise.

After RQ-factorization, we need to:

- Fix **t**.
- Set element (3,3) in **A** to 1.
- **R** should have det(R)=1 (no mirroring)

Matlab code (written by Björn Johansson)

```
function [K,R,t] = P2KRt(P)

% [K,R,t] = P2KRt(P)
% Computes camera matrix K, rotation R, and translation t
% fron projection matrix P. Relation:
%          P ~ K[R t]
%   P - 3/4 projection matrix
%   K - 3/3 camera matrix
%   R - 3/3 rotation matrix
%   t - 3/1 translation vector

[K,R] = rq(P(:,1:3));
t = inv(K)*P(:,4);
K = K/K(3,3);
```

Note:
A is now denoted **K**
C is denoted **P**

Matlab code

```
% K should have positive sign along the diagonal
D = diag(sign(diag(K)));
K = K*D;
R = D*R;
t = D*t;

% R should have det(R)=1 (no mirroring)
t = det(R)*t;
R = det(R)*R;
```

Matlab code

```
function [R,Q] = rq(A)
% [R,Q] = rq(A)
% Orthogonal-triangular decomposition, A = R*Q, where
% R is an upper triangular matrix and
% Q is an orthogonal matrix.
A = A';
A = A(end:-1:1,end:-1:1);
[Q,R] = qr(A);
R = R';R = R(end:-1:1,end:-1:1);
Q = Q';Q = Q(end:-1:1,end:-1:1);
```