

# TSBB21, Lecture 4b

## Camera calibration 1

p. 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

Maria Magnusson, CVL, Dept. of Electrical Engineering, Linköping University



# Transformation

## with homogenous matrices

p. 2

- 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

p. 3

Translation

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

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

p. 4

Scaling

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

Eq. (3)

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

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

Eq. (10)

General skewing

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

Eq. (11)



## Homogeneous matrices for rotation

p. 5

Rotation with the angle  $\theta$  around the x-axis

Eq. (7)

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

Eq. (8)

Rotation with the angle  $\theta$  around the y-axis

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

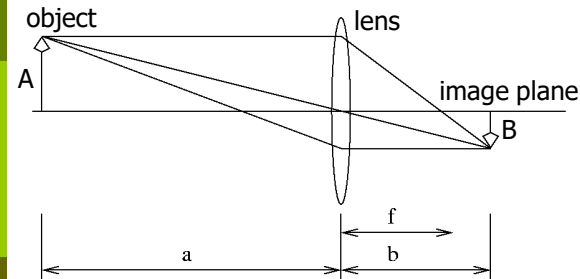
Rotation with the angle  $\theta$  around the z-axis

Eq. (9)

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

## The Lens law (repetition)

p. 6



The lens law:

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

where  $f$  is the focal length

Size relations:

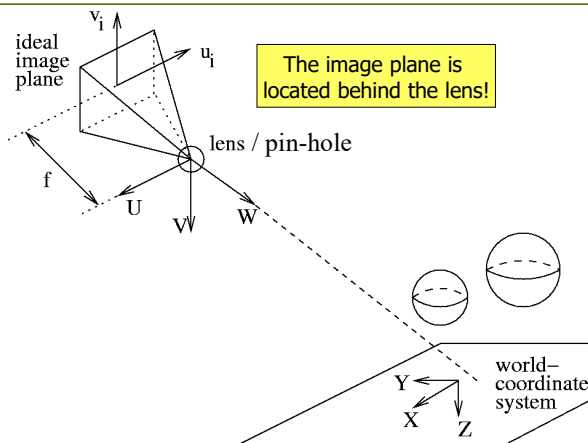
$$\frac{A}{a} = \frac{B}{b} \approx \frac{B}{f}$$

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  $a \gg b \Rightarrow b \approx f$ .

## The pinhole camera model, real geometry

p. 7

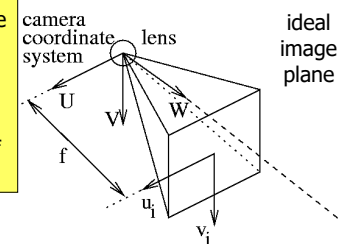


The image plane is located behind the lens!

## The pinhole camera model, mirrored

p. 8

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



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.

Relation between the coordinates of the two coordinate systems:

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

Fig. 2

Eq. (12)

## Technique to express perspective transformation with vectors <sup>p. 9</sup>

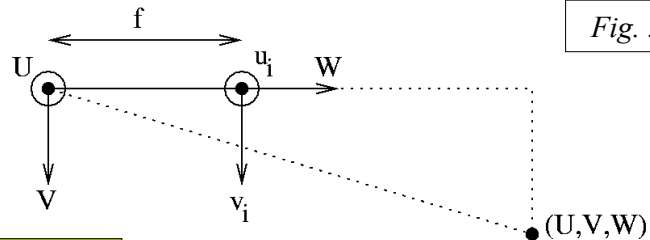


Fig. 3

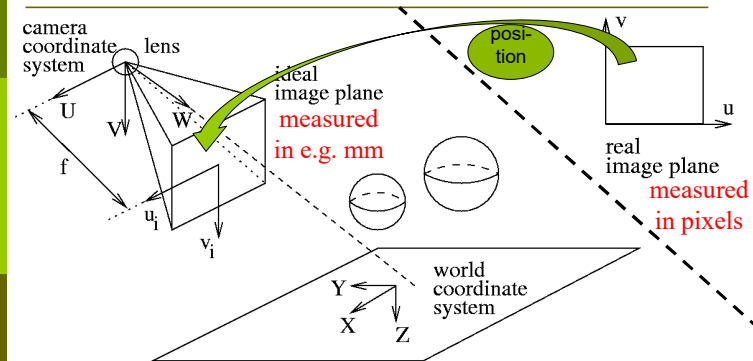
Uniform triangles gives:

$u_n$  and  $v_n$  are the normalized image coordinates

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

Eq. (14)

## Relation between the ideal image plane and the real image plane <sup>p. 10</sup>



Eq. (15)

$$(u, v, 1)^T = A \cdot \begin{pmatrix} \frac{u_i}{f}, \frac{v_i}{f}, 1 \end{pmatrix}^T$$

## Relation between world coordinates and real image coordinates <sup>p. 11</sup>

Relation between the coordinate systems:

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

Eq. (12)

Relation between the image planes:

$$(u, v, 1)^T = A \cdot \begin{pmatrix} \frac{u_i}{f}, \frac{v_i}{f}, 1 \end{pmatrix}^T$$

Eq. (15)

Relation between world coordinates and real image coordinates:

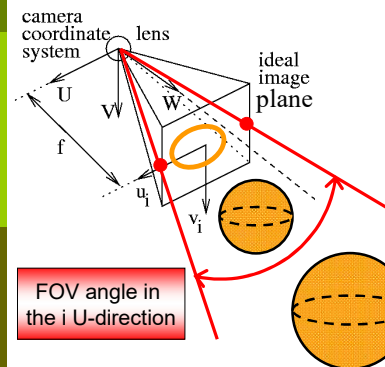
$$(u, v, 1)^T \sim s(u, v, 1)^T = A[Rt] \cdot (X, Y, Z, 1)^T$$

equivalence

Eq. (18)

Note that  $s$  replaces  $W$  as "perspective projection parameter"

## Unambiguity, field of view (FOV) and resolution <sup>p. 12</sup>



- 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:

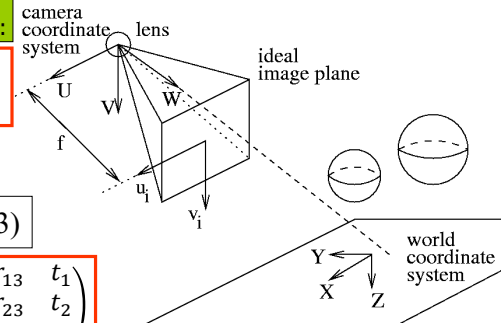
$$(U, V, W)^T = [\mathbf{Rt}] \cdot (X, Y, Z, 1)^T$$

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  
 $\mathbf{R}$ : the rotation of the camera in relation to the world



## Inner parameters

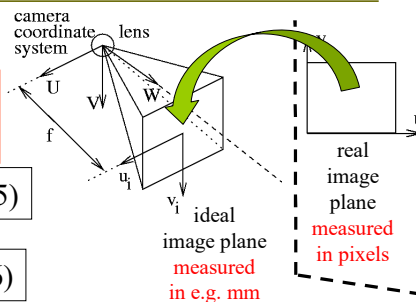
Relation between the image planes:

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

Eq. (15)

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

Eq. (16)

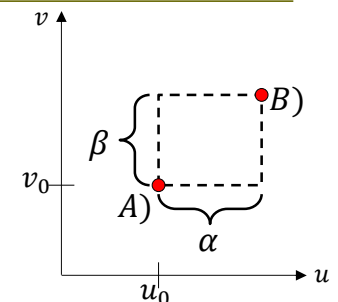


$\beta$ : scaling in the v-direction  
 $\alpha$ : scaling in the u-direction  
 $\gamma$ : skewing (lack of orthogonality between horizontal and vertical axes) (often close to 0)  
 $(u_0, v_0)$ : the cross-section between the optical axis and the real image plane

## Inner parameters, ex) with $\gamma=0$

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

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

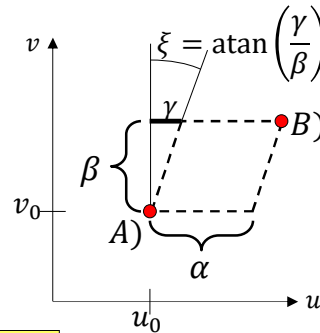


$(u_0, v_0)$ : the cross-section between the optical axis and the real image plane, the image center, the principal point.  
 $\alpha$  and  $\beta$  denotes the scaling in the u- and v-direction, respectively.  
 If  $\alpha = \beta$ , the pixels are quadratic.  
 If  $\alpha \neq \beta$ , the pixels 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) | $(\alpha + \gamma + u_0, \beta + v_0)$ | $(f, f)$     |



$\gamma$  is the skewing parameter  
 $\xi = \arctan(\gamma/\beta)$  gives an angular measurement  
 $\xi$  is normally small, i.e. close to 0 degrees



## 3D calibration of a camera

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

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

We will first determine  $\mathbf{C}$ , only.  
 Later, we will learn how to determine  $\mathbf{A}$ ,  $\mathbf{R}$  and  $\mathbf{t}$ .

Depending on the variable  $s$ ,  
 $\mathbf{C}$  can only be determined up to  
 a scale factor, say  $\lambda$ .

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

We have now two possibilities,  
 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.)

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

The matrix  $\mathbf{C}$  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)

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

Eq. (19)

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

$$\begin{pmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -u_1 X_1 & -u_1 Y_1 & -u_1 Z_1 \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -v_1 X_1 & -v_1 Y_1 & -v_1 Z_1 \\ X_2 & Y_2 & Z_2 & 1 & 0 & 0 & 0 & 0 & -u_2 X_2 & -u_2 Y_2 & -u_2 Z_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & X_N & Y_N & Z_N & 1 & -u_N X_N & -u_N Y_N & -u_N Z_N \end{pmatrix} \begin{pmatrix} c_{11} \\ c_{12} \\ c_{13} \\ c_{33} \end{pmatrix} = \begin{pmatrix} u_1 \\ v_1 \\ u_2 \\ \vdots \\ v_N \end{pmatrix} \quad \text{Eq. (22)}$$

11 equations give that at least 6 point-pairs ("5½") is needed to determine  $\mathbf{C}$



## Show Eq. (21)

Show Eq. (21)

We have measured the point  $(X_i, Y_i, Z_i)$  in the world.

It corresponds to  $(u_i, v_i)$  in the image.

$$(18, 19) \Rightarrow s \begin{pmatrix} u_i \\ v_i \\ 1 \end{pmatrix} = \begin{pmatrix} C_{11} & C_{12} & C_{13} & C_{14} \\ C_{21} & C_{22} & C_{23} & C_{24} \\ C_{31} & C_{32} & C_{33} & 1 \end{pmatrix} \begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix}$$

$$s u_i = C_{11} X_i + C_{12} Y_i + C_{13} Z_i + C_{14} \quad (a)$$

$$s v_i = \dots \quad (b)$$

$$s = C_{31} X_i + C_{32} Y_i + C_{33} Z_i + 1 \quad (c)$$

$$(a, c) \Rightarrow u_i = C_{11} X_i + C_{12} Y_i + C_{13} Z_i + C_{14} - C_{31} X_i u_i - C_{32} Y_i u_i - C_{33} Z_i u_i$$

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:

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

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!

$\mathbf{D}^+$  is pinv in Matlab

$$\begin{aligned} \mathbf{D} \cdot \mathbf{c} &= \mathbf{f} \\ \mathbf{D}^T \mathbf{D} \cdot \mathbf{c} &= \mathbf{D}^T \mathbf{f} \\ \mathbf{c} &= (\mathbf{D}^T \mathbf{D})^{-1} \mathbf{D}^T \mathbf{f} \\ \mathbf{c} &= \mathbf{D}^+ \mathbf{f} \end{aligned}$$

Eq. (23)

$\mathbf{D}^+$  is the so called pseudo-inverse of  $\mathbf{D}$ .  
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  $\mathbf{C}$  is kept as:

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

- 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:

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

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

$$\begin{pmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -u_1 X_1 & -u_1 Y_1 & -u_1 Z_1 & -u_1 \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -v_1 X_1 & -v_1 Y_1 & -v_1 Z_1 & -v_1 \\ X_2 & Y_2 & Z_2 & 1 & 0 & 0 & 0 & 0 & -u_2 X_2 & -u_2 Y_2 & -u_2 Z_2 & -u_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & X_N & Y_N & Z_N & 1 & -v_N X_N & -v_N Y_N & -v_N Z_N & -v_N \end{pmatrix} \begin{pmatrix} C_{11} \\ C_{12} \\ C_{13} \\ C_{14} \\ C_{21} \\ C_{22} \\ C_{23} \\ C_{24} \\ C_{31} \\ C_{32} \\ C_{33} \\ C_{34} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Matlab solution:

$$[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \text{svd}(\mathbf{D});$$

$$\mathbf{c} = \mathbf{V}(:, 12);$$

$\mathbf{c}$  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.

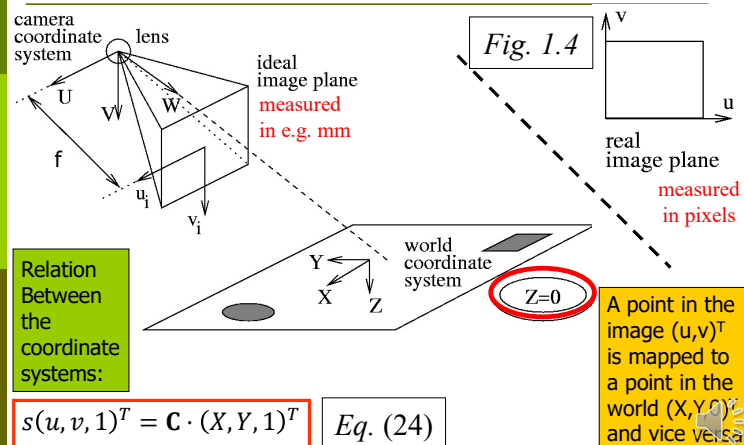


## Using the calibrated camera

- 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 give 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



## Inhomogeneous solution of a homography

$$s(u, v, 1)^T = C \cdot (X, Y, 1)^T \quad \text{Eq. (24)}$$

$$C = \begin{pmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & 1 \end{pmatrix} \quad \text{Eq. (25)}$$

Set:

$$c = (c_{11}, c_{12}, c_{13}, c_{21}, c_{22}, c_{23}, c_{31}, c_{32}) \quad \text{Eq. (26)} \quad \text{Eq. (27)}$$

$$D \cdot c = \begin{pmatrix} X_1 & Y_1 & 1 & 0 & 0 & 0 & -u_1 X_1 & -u_1 Y_1 \\ 0 & 0 & 0 & X_1 & Y_1 & 1 & -v_1 X_1 & -v_1 Y_1 \\ X_2 & Y_2 & 1 & 0 & 0 & 0 & -u_2 X_2 & -u_2 Y_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & X_N & Y_N & 1 & -v_N X_N & -v_N Y_N \end{pmatrix} \begin{pmatrix} c_{11} \\ c_{12} \\ c_{13} \\ c_{21} \\ c_{22} \\ c_{23} \\ c_{31} \\ c_{32} \end{pmatrix} = \begin{pmatrix} u_1 \\ v_1 \\ u_2 \\ \vdots \\ v_N \end{pmatrix} = f$$

Solution as before:  $c = D^+ f$ Matlab solution:  $c = \text{pinv}(D) * f$ 

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



## Homogeneous solution of a homography

p. 29

**Note:**  
Hartley normalization  
(see a previous slide)  
may improve performance!

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

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

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

p. 30

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

p. 31

- If  $\mathbf{C}$  is not at infinity then we can always find a unique decomposition of  $\mathbf{C}$  into its internal  $\mathbf{A}$  and external  $[\mathbf{Rt}]$  parameters. This decomposition is referred to as camera resectioning.
- $\mathbf{A}$  is an **upper triangular** 3x3 matrix
- $\mathbf{R}$  is a **rotational matrix**, which describes rotations around the X, Y- and Z-axes.
- $\mathbf{R}$  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  $\mathbf{I}$  is the identity matrix.
- $\mathbf{t}$  is a translation vector, which describes a translation along the X, Y- and Z-axes.



## QR- and RQ-factorization

p. 32

- QR-factorization decomposes a matrix  $\mathbf{B}$  into an orthogonal matrix  $\mathbf{Q}$  multiplied by an upper (or right) triangular matrix  $\mathbf{R}$ .
- Matlab command: `[Q, R] = qr(B);`
- $\mathbf{B}$  and  $\mathbf{Q}$  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:  
 $\mathbf{R}$  has different meanings!  
The triangular  $\mathbf{R}$  is marked with turquoise.





## After RQ-factorization, we need to:

p. 33

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



## Matlab code (written by Björn Johansson)

p. 34

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

% [K,R,t] = P2KRt(P)
% Computes camera matrix K, rotation R, and translation t
% from 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

p. 35

```
% 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

p. 36

```
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);
```

