



Computer Vision

03 – Image formation, part 2

doc. dr. Janez Perš
(with contributions by prof. Stanislav Kovačič)

Laboratory for Machine Intelligence
Faculty of Electrical Engineering
University of Ljubljana

Announcements (2/2)

- E-classroom
 - <http://e.fe.uni-lj.si>
 - Lecture slides, lab tasks
 - Lab tasks will be available *only* through Echo
- Labs start(ed) this week!
 - Lab *attendance* mandatory (prerequisite for exams)
 - *Completion of lab work* is mandatory (prerequisite for exams)

Quick recap of the previous week

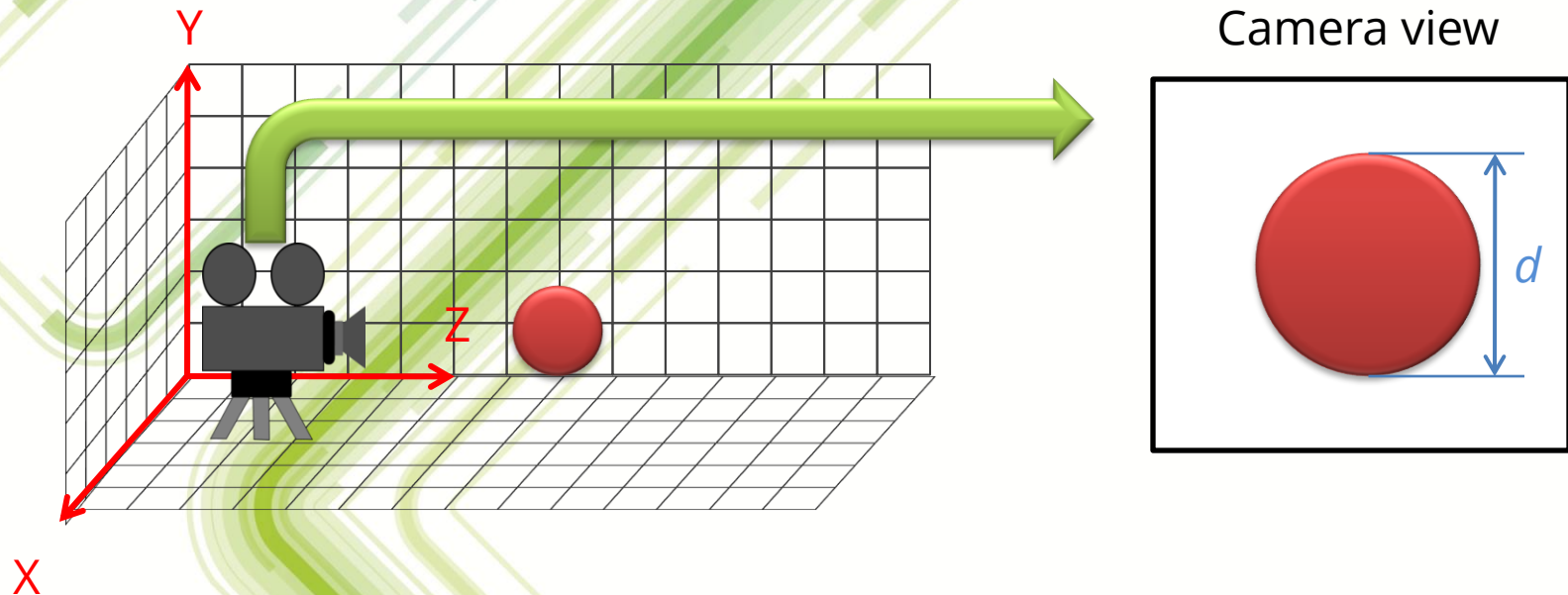
- Camera model
 - Yes, the *mathematical* model
 - Camera coordinate system
 - World coordinate system
- Camera calibration
 - Extrinsic parameters
 - Intrinsic parameters
 - Camera calibration via DLT
- Camera sensor
 - Image formation, pixels, pixel formats

Outline

- Perspective projection revisited
 - The “Problem of $1/z$ ”
 - Homogeneous coordinates to the rescue
- Projective geometry revisited
- Camera model revisited
 - Camera calibration matrix, projection matrix
 - we will build both step by step
- Any experience with computer graphics?
 - Homogeneous coordinates are the norm in CG
- Lens distortions

The “Problem of $1/Z$ ”

- Imagine an object in 3D Euclidean space
 - moving away from the observer!
 - What does the camera see?

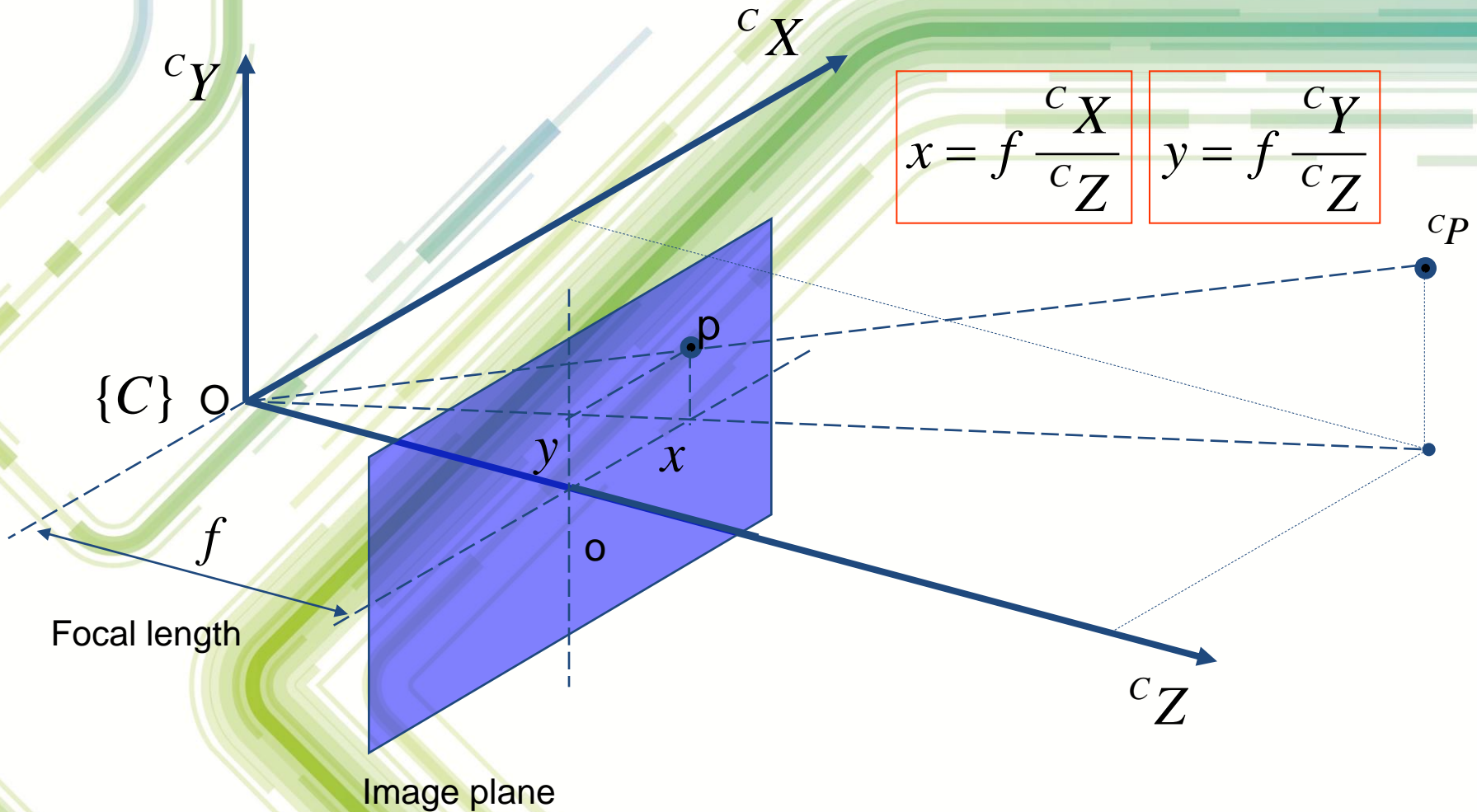


- The size of the projected image (d) is proportional to $1/z$!
- $d = 0$ when $Z = \infty$

The “Problem of $1/Z$ ”

- Let's repeat!
 - We have Euclidean 3D coordinate system (X,Y,Z)
 - The object moves along Z dimension away from the camera
 - The size of the projection camera sees is proportional to $1/z$
- This appears to be a big problem
 - Simple motion along coordinate axis
 - And yet the projection results in the non-linear relationship ($1/Z$) between the object coordinate Z and the projection size d !
 - We have non-linear transformation
 - Should we forget about using linear algebra on this?

Perspective projection revisited (1/4)



Perspective projection revisited (2/4)

$$x = f \frac{{}^c X}{{}^c Z}$$



$${}^c Z \cdot x = f \cdot {}^c X$$



$$w \cdot x = f \cdot {}^c X$$



$$w \cdot x = f \cdot {}^c X + 0 \cdot {}^c Y + 0 \cdot {}^c Z$$

$$y = f \frac{{}^c Y}{{}^c Z}$$



$${}^c Z \cdot y = f \cdot {}^c Y$$



$$w \cdot y = f \cdot {}^c Y$$



$$w \cdot y = 0 \cdot {}^c X + f \cdot {}^c Y + 0 \cdot {}^c Z$$

$$w = {}^c Z$$



$$w \cdot 1 = {}^c Z$$

$$w \cdot x = f \cdot {}^c X + 0 \cdot {}^c Y + 0 \cdot {}^c Z$$

$$w \cdot y = 0 \cdot {}^c X + f \cdot {}^c Y + 0 \cdot {}^c Z$$

$$w \cdot 1 = 0 \cdot {}^c X + 0 \cdot {}^c Y + 1 \cdot {}^c Z$$

Perspective projection revisited (3/4)

$$w \cdot x = f \cdot {}^cX + 0 \cdot {}^cY + 0 \cdot {}^cZ$$

$$w \cdot y = 0 \cdot {}^cX + f \cdot {}^cY + 0 \cdot {}^cZ$$

$$w \cdot 1 = 0 \cdot {}^cX + 0 \cdot {}^cY + 1 \cdot {}^cZ$$



Homogeneous vector

$$w \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^cX \\ {}^cY \\ {}^cZ \end{bmatrix}$$

Matrix of camera internal parameters
(only one parameter, i.e. f , so far)

For $f = 1$, we say that the image plane coordinates are normalized, or in a *'canonical'* form.

Perspective projection revisited (4/4)

$$w \cdot x = f \cdot {}^cX + 0 \cdot {}^cY + 0 \cdot {}^cZ$$

$$w \cdot y = 0 \cdot {}^cX + f \cdot {}^cY + 0 \cdot {}^cZ$$


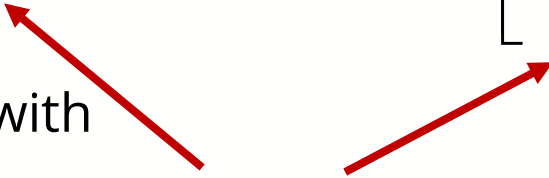
$$w \cdot 1 = 0 \cdot {}^cX + 0 \cdot {}^cY + 1 \cdot {}^cZ$$

$$w \cdot x = f \cdot {}^cX + 0 \cdot {}^cY + 0 \cdot {}^cZ + 0 \cdot 1$$

$$w \cdot y = 0 \cdot {}^cX + f \cdot {}^cY + 0 \cdot {}^cZ + 0 \cdot 1$$

$$w \cdot 1 = 0 \cdot {}^cX + 0 \cdot {}^cY + 1 \cdot {}^cZ + 0 \cdot 1$$

- This still describes transformation from 3D to 2D!
- Additional dimensions (3rd on the left, 4th on the right) allow us to handle projection using linear algebra.
- We will illustrate this later!
- We will also “fill” zeros in the matrix with other camera parameters


$$w \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} {}^cX \\ {}^cY \\ {}^cZ \\ 1 \end{bmatrix}$$


Homogeneous vectors

Homogeneous vectors refresher (1/3)

Homogeneous vectors, and coordinates, are used frequently in computer vision, computer graphics, robotics, ... due to their *mathematical convenience*.

Homogeneous coordinates can be used to represent vectors in 2D or 3D space, or in space of any dimension simply by extending or ,augmenting' the vector by one additional coordinate or ,dimension', typically of value one.

Coordinates x, y (i.e., components of a vector p in 2D Euclidean space) can be written in homogenous form as

$$\tilde{p} = \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{w} \end{bmatrix} = \tilde{w} \begin{bmatrix} \tilde{x} / \tilde{w} \\ \tilde{y} / \tilde{w} \\ 1 \end{bmatrix} = \tilde{w} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \quad \tilde{p} \in \mathcal{P}^2 \text{ 2D projective space}$$

Thus, to go from homogeneous back to real coordinates we simply divide the first two components with the third one.

Homogeneous vectors refresher (2/3)

Converting from real to homogeneous coordinates (in 2D and 3D)

$$\mathbf{p} = \begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow \tilde{\mathbf{p}} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

homogeneous image
coordinates

$$\mathbf{p} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \Rightarrow \tilde{\mathbf{p}} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

homogeneous scene
coordinates

Converting from homogeneous to real coordinates (in 2D and 3D)

$$\tilde{\mathbf{p}} = \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{w} \end{bmatrix} \Rightarrow \mathbf{p} = \begin{bmatrix} \tilde{x} / \tilde{w} \\ \tilde{y} / \tilde{w} \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\tilde{\mathbf{p}} = \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \\ \tilde{w} \end{bmatrix} \Rightarrow \mathbf{p} = \begin{bmatrix} \tilde{x} / \tilde{w} \\ \tilde{y} / \tilde{w} \\ \tilde{z} / \tilde{w} \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Quite often (if no ambiguity occurs) we can omit the tilde sign

Homogeneous vectors refresher (3/3)

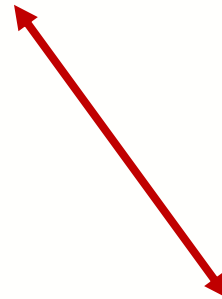
Important property: homogeneous vectors that differ only in scale are equivalent. Thus, they represent *the same real point*.

$$\tilde{\mathbf{p}}_b = k \cdot \tilde{\mathbf{p}}_a \cong \tilde{\mathbf{p}}_a, \quad k \neq 0$$

$$\tilde{\mathbf{p}}_a = \begin{bmatrix} \tilde{x}_a \\ \tilde{y}_a \\ \tilde{w}_a \end{bmatrix} = \tilde{w}_a \begin{bmatrix} \tilde{x}_a / \tilde{w}_a \\ \tilde{y}_a / \tilde{w}_a \\ 1 \end{bmatrix} = \tilde{w}_a \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Rightarrow \mathbf{p} = \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\tilde{\mathbf{p}}_b = \begin{bmatrix} \tilde{x}_b \\ \tilde{y}_b \\ \tilde{w}_b \end{bmatrix} = k \cdot \tilde{\mathbf{p}}_a = k \begin{bmatrix} \tilde{x}_a \\ \tilde{y}_a \\ \tilde{w}_a \end{bmatrix} = \begin{bmatrix} k\tilde{x}_a \\ k\tilde{y}_a \\ k\tilde{w}_a \end{bmatrix}$$

$$\begin{bmatrix} k\tilde{x}_a \\ k\tilde{y}_a \\ k\tilde{w}_a \end{bmatrix} = k \cdot \tilde{w}_a \begin{bmatrix} k\tilde{x}_a / k\tilde{w}_a \\ k\tilde{y}_a / k\tilde{w}_a \\ 1 \end{bmatrix} = \tilde{w}_b \begin{bmatrix} \tilde{x}_a / \tilde{w}_a \\ \tilde{y}_a / \tilde{w}_a \\ 1 \end{bmatrix} = \tilde{w}_b \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Rightarrow \mathbf{p} = \begin{bmatrix} x \\ y \end{bmatrix}$$



Perspective projection again (1/3)

$$x = f \frac{cX}{cZ}$$

$$y = f \frac{cY}{cZ}$$

Now it's time to discretize the image plane.

$$\lambda_u(u - u_0) = x$$

$$\lambda_v(v - v_0) = y$$

$$\lambda_u(u - u_0) = f \frac{cX}{cZ} \Rightarrow u - u_0 = \frac{f}{\lambda_u} \cdot \frac{cX}{cZ} = f_u \frac{cX}{cZ}$$

$$\lambda_v(v - v_0) = f \frac{cY}{cZ} \Rightarrow v - v_0 = \frac{f}{\lambda_v} \cdot \frac{cY}{cZ} = f_v \frac{cY}{cZ}$$

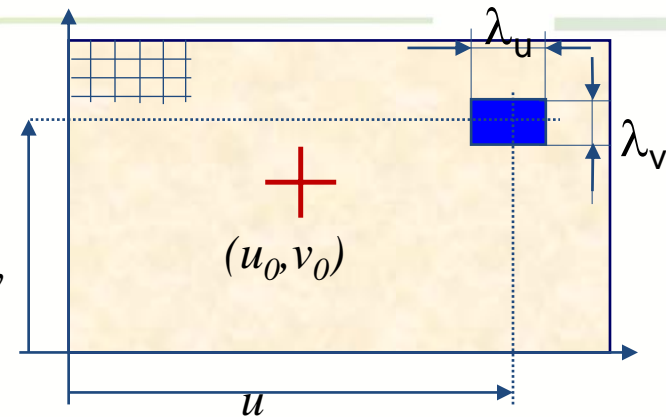
$$f_u = f / \lambda_u$$

$$f_v = f / \lambda_v$$

Effective focal lengths in horizontal and vertical direction, expressed in pixels.

$$(u_0, v_0)$$

Principal point: intersection of image plane and optical axis



Perspective projection again (2/3)

$${}^cZ \cdot (u - u_0) = f_u \cdot {}^cX$$



$${}^cZ \cdot u = f_u \cdot {}^cX + u_0 \cdot {}^cZ$$



$$w \cdot u = f_u \cdot {}^cX + u_0 \cdot {}^cZ$$



$$w \cdot u = f_u \cdot {}^cX + 0 \cdot {}^cY + u_0 \cdot {}^cZ$$

$${}^cZ \cdot (v - v_0) = f_v \cdot {}^cY$$



$${}^cZ \cdot v = f_v \cdot {}^cY + v_0 \cdot {}^cZ$$



$$w \cdot v = f_v \cdot {}^cY + v_0 \cdot {}^cZ$$



$$w \cdot v = 0 \cdot {}^cX + f_v \cdot {}^cY + v_0 \cdot {}^cZ$$

$$w = {}^cZ$$



$$w \cdot \mathbf{1} = {}^cZ$$

$$w \cdot u = f_u \cdot {}^cX + 0 \cdot {}^cY + u_0 \cdot {}^cZ$$

$$w \cdot v = 0 \cdot {}^cX + f_v \cdot {}^cY + v_0 \cdot {}^cZ$$

$$w \cdot \mathbf{1} = 0 \cdot {}^cX + 0 \cdot {}^cY + 1 \cdot {}^cZ$$



$$w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^cX \\ {}^cY \\ {}^cZ \end{bmatrix}$$

Perspective projection again (2/3)

Additional parameter: *skew*
In reality very small



We already have discretized coordinates. Finally, we add skew

$$w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & \gamma & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^c X \\ {}^c Y \\ {}^c Z \end{bmatrix}$$

Calibration matrix - K

effective focal lengths skew principal point

$$w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & \gamma & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^cX \\ {}^cY \\ {}^cZ \end{bmatrix} = \mathbf{K} \begin{bmatrix} {}^cX \\ {}^cY \\ {}^cZ \end{bmatrix}$$

- This matrix
 - is the matrix of *camera internal parameters*,
 - is often called *calibration matrix*, or *camera parameters matrix*,
 - and denoted K.

Projection matrix

To obtain homogeneous coordinates on both sides, we simply add one more column of zeros.

$$w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & \gamma & u_0 & 0 \\ 0 & f_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} {}^cX \\ {}^cY \\ {}^cZ \\ 1 \end{bmatrix}$$

Or, equivalently, we introduce one more 3x4 matrix, that annihilates the appended 1.

$$w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & \gamma & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} {}^cX \\ {}^cY \\ {}^cZ \\ 1 \end{bmatrix}$$

Yet another way to look at it is to take the 3 x 4 matrix above and “decompose” it into a product of 3 x 3 matrix and 3 x 4 matrix (below).

Projection matrix

Until now we have ignored the world coordinate system.
Or, equivalently, we have assumed that the camera coordinate system coincides with the world coordinate system.

Therefore, ${}^cP = P$, or component-wise $[{}^cX \ {}^cY \ {}^cZ]^T = [X \ Y \ Z]^T$

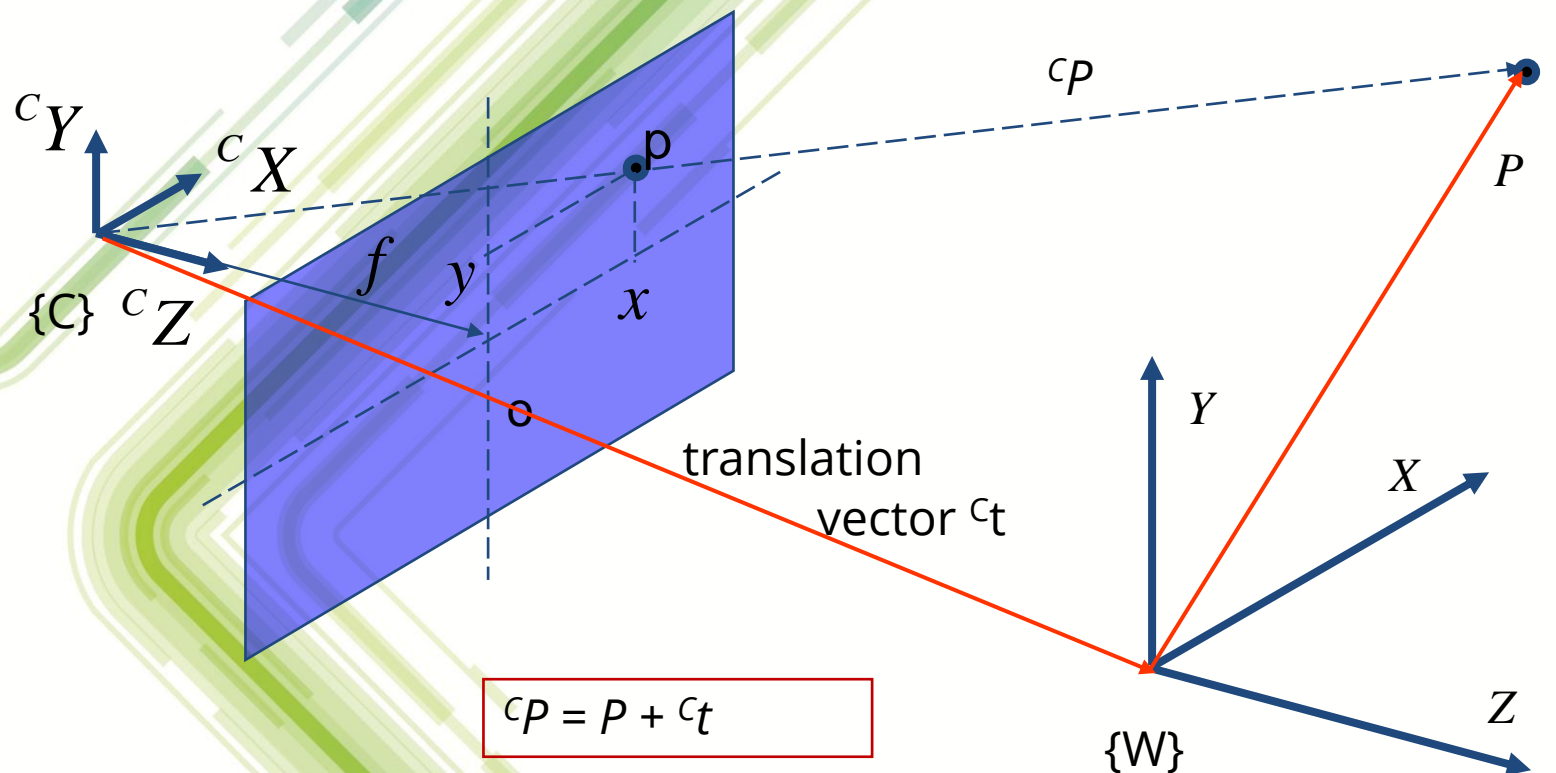
$$w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & \gamma & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{K} [\mathbf{I}_{3 \times 3} \mid \mathbf{0}_{3 \times 1}] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

This is the mapping from 3D world to the 2D image, under the assumption that camera c.s. coincides with world c.s.

Introducing translation (1/2)

Let's assume that the world coordinate system (a $\{W\}$ "frame") is moved, i.e. translated for t with respect to the camera frame $\{C\}$.

Note that we could also think of moving camera for $-t$ w.r.t. $\{W\}$.



Introducing translation (2/2)

$$\begin{bmatrix} {}^cX \\ {}^cY \\ {}^cZ \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} {}^ct_x \\ {}^ct_y \\ {}^ct_z \end{bmatrix} \Rightarrow \begin{aligned} {}^cX &= 1 \cdot X + 0 \cdot Y + 0 \cdot Z + {}^ct_x \cdot 1 \\ {}^cY &= 0 \cdot X + 1 \cdot Y + 0 \cdot Z + {}^ct_y \cdot 1 \\ {}^cZ &= 0 \cdot X + 0 \cdot Y + 1 \cdot Z + {}^ct_z \cdot 1 \end{aligned}$$



$$\begin{bmatrix} {}^cX \\ {}^cY \\ {}^cZ \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & {}^ct_x \\ 0 & 1 & 0 & {}^ct_y \\ 0 & 0 & 1 & {}^ct_z \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Translation in homogeneous coordinates

$$\begin{bmatrix} {}^cX \\ {}^cY \\ {}^cZ \\ 1 \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} + \begin{bmatrix} {}^c t_x \\ {}^c t_y \\ {}^c t_z \\ 1 \end{bmatrix} \Rightarrow \begin{aligned} {}^cX &= 1 \cdot X + 0 \cdot Y + 0 \cdot Z + {}^c t_x \cdot 1 \\ {}^cY &= 0 \cdot X + 1 \cdot Y + 0 \cdot Z + {}^c t_y \cdot 1 \\ {}^cZ &= 0 \cdot X + 0 \cdot Y + 1 \cdot Z + {}^c t_z \cdot 1 \\ 1 &= 0 \cdot X + 0 \cdot Y + 0 \cdot Z + 1 \end{aligned}$$



$$\begin{bmatrix} {}^cX \\ {}^cY \\ {}^cZ \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & {}^c t_x \\ 0 & 1 & 0 & {}^c t_y \\ 0 & 0 & 1 & {}^c t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

The nice thing about homogenous vectors is that translation turns into linear *matrix transformation*, i.e. *matrix multiplication*.

Translation in abridged notation

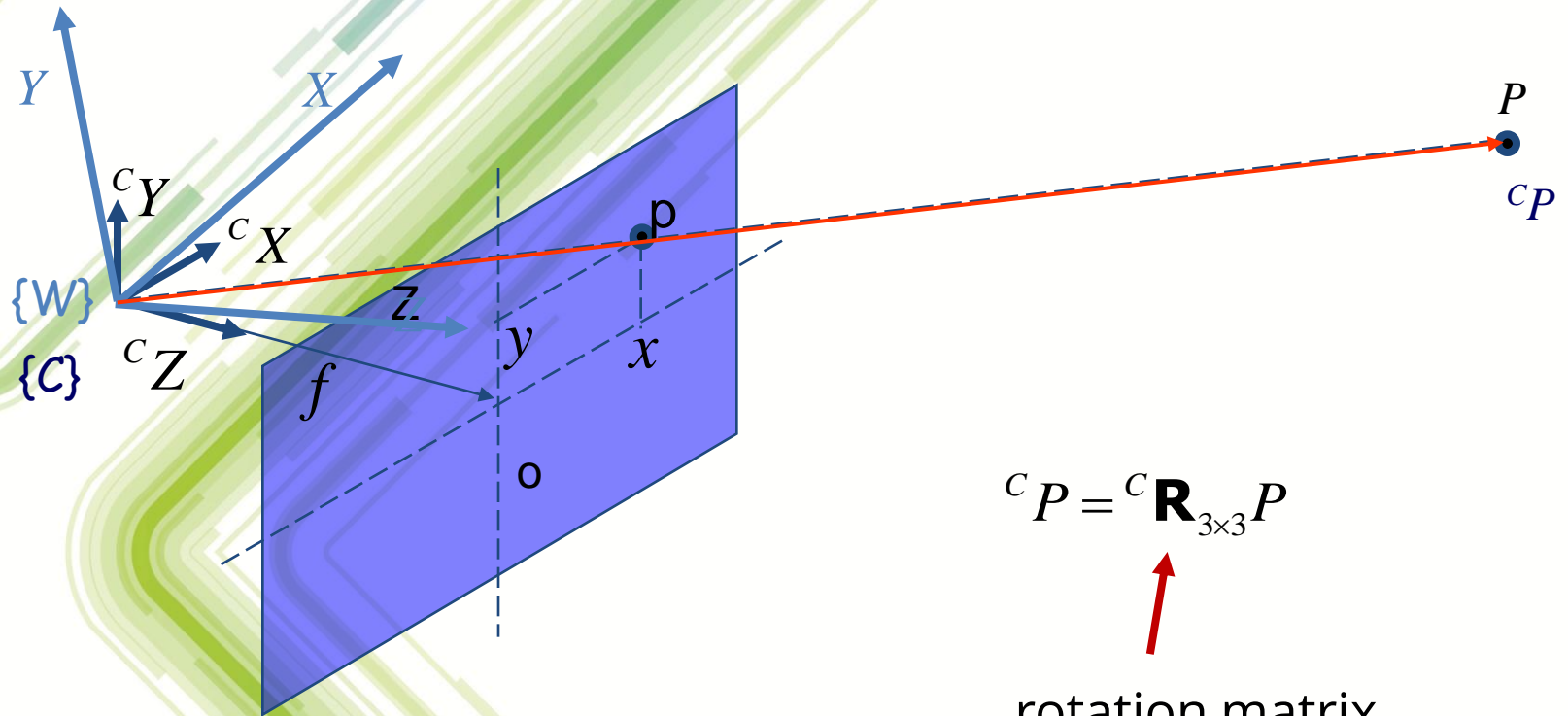
$${}^C \tilde{P} = \begin{bmatrix} \mathbf{I}_{3 \times 3} & {}^C \mathbf{t}_{3 \times 1} \\ \mathbf{0}_{1 \times 3}^T & 1 \end{bmatrix} \tilde{P} = {}^C \tilde{\mathbf{T}}_W {}^W \tilde{P}$$

- This mapping takes us from $\{W\}$ to $\{C\}$.
 - Note that translation vector \mathbf{t} is defined w.r.t. $\{C\}$.
- From time to time we will omit the leading superscripts.

Introducing rotation

Let us concentrate on rotation, and ignore translation for a moment.

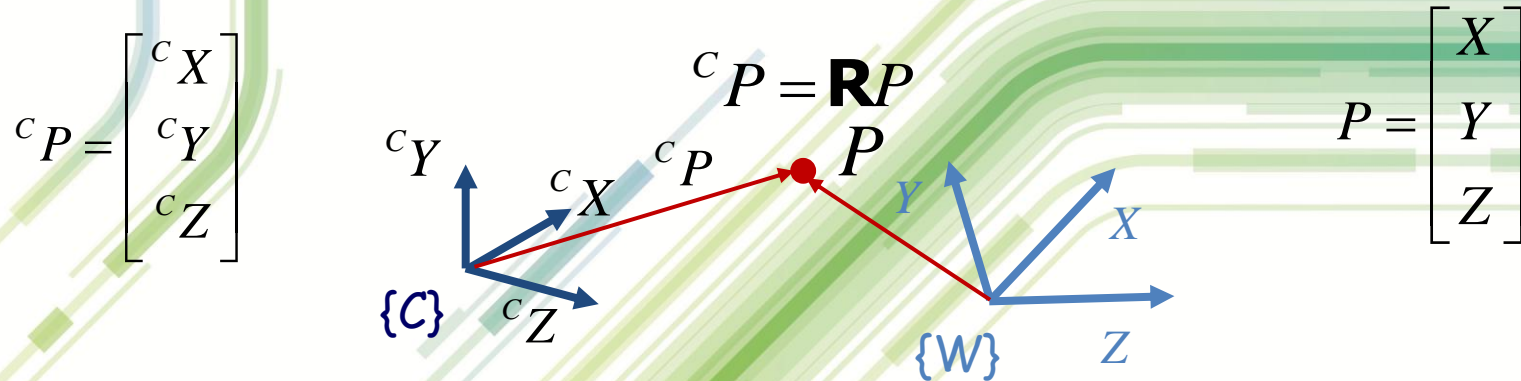
In our case coordinate frame $\{W\}$ is rotated w.r.t. $\{C\}$.



$${}^cP = {}^c\mathbf{R}_{3 \times 3} P$$

rotation matrix

Linear algebra refresher (1/4)



$b_C = \{e_{X_c}, e_{Y_c}, e_{Z_c}\}$ Orthogonal unit vectors along $\{C\}$ axes, a ,basis'

$b = \{e_X, e_Y, e_Z\}$ Orthogonal unit vectors along $\{W\}$ axes, a ,basis'

$${}^cP = {}^cX \cdot e_{X_c} + {}^cY \cdot e_{Y_c} + {}^cZ \cdot e_{Z_c}$$

$$P = X \cdot e_X + Y \cdot e_Y + Z \cdot e_Z$$

$${}^cP = P$$

$${}^cX \cdot e_{X_c} + {}^cY \cdot e_{Y_c} + {}^cZ \cdot e_{Z_c} = X \cdot e_X + Y \cdot e_Y + Z \cdot e_Z$$

Linear algebra refresher (2/4)

$${}^cX \cdot e_{X_c} + {}^cY \cdot e_{Y_c} + {}^cZ \cdot e_{Z_c} = X \cdot e_X + Y \cdot e_Y + Z \cdot e_Z \quad / \text{ multiply by } e_{X_c}, e_{Y_c}, e_{Z_c}$$

$${}^cX \cdot (\overset{1}{e_{X_c} \cdot e_{X_c}}) + {}^cY \cdot (\overset{0}{e_{Y_c} \cdot e_{X_c}}) + {}^cZ \cdot (\overset{0}{e_{Z_c} \cdot e_{X_c}}) = X \cdot (e_X \cdot e_{X_c}) + Y \cdot (e_Y \cdot e_{X_c}) + Z \cdot (e_Z \cdot e_{X_c})$$

$${}^cX \cdot (\overset{0}{e_{X_c} \cdot e_{Y_c}}) + {}^cY \cdot (\overset{1}{e_{Y_c} \cdot e_{Y_c}}) + {}^cZ \cdot (\overset{0}{e_{Z_c} \cdot e_{Y_c}}) = X \cdot (e_X \cdot e_{Y_c}) + Y \cdot (e_Y \cdot e_{Y_c}) + Z \cdot (e_Z \cdot e_{Y_c})$$

$${}^cX \cdot (\overset{0}{e_{X_c} \cdot e_{Z_c}}) + {}^cY \cdot (\overset{0}{e_{Y_c} \cdot e_{Z_c}}) + {}^cZ \cdot (\overset{1}{e_{Z_c} \cdot e_{Z_c}}) = X \cdot (e_X \cdot e_{Z_c}) + Y \cdot (e_Y \cdot e_{Z_c}) + Z \cdot (e_Z \cdot e_{Z_c})$$

$$\begin{bmatrix} {}^cX \\ {}^cY \\ {}^cZ \end{bmatrix} = \begin{bmatrix} (e_X \cdot e_{X_c}) & (e_Y \cdot e_{X_c}) & (e_Z \cdot e_{X_c}) \\ (e_X \cdot e_{Y_c}) & (e_Y \cdot e_{Y_c}) & (e_Z \cdot e_{Y_c}) \\ (e_X \cdot e_{Z_c}) & (e_Y \cdot e_{Z_c}) & (e_Z \cdot e_{Z_c}) \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

The columns are nothing but projections of basis vectors (vector components)

$$b = \{e_X, e_Y, e_Z\} \text{ onto } b_C = \{e_{X_c}, e_{Y_c}, e_{Z_c}\}$$

Linear algebra refresher (3/4)

$$\begin{bmatrix} {}^cX \\ {}^cY \\ {}^cZ \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = {}^c\mathbf{R} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Therefore, \mathbf{R} is a linear mapping that takes any vector \mathbf{P} expressed in $\{W\}$ and maps it into vector ${}^c\mathbf{P}$ expressed in $\{C\}$.

In case of rotation, \mathbf{R} is orthogonal (preserves distances).

In homogeneous coordinates, and in case of pure rotation:

$$\begin{bmatrix} {}^cX \\ {}^cY \\ {}^cZ \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} {}^c\mathbf{R}_{3 \times 3} & \mathbf{0}_{3 \times 1} \\ \mathbf{0}_{1 \times 3}^T & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = {}^c\tilde{\mathbf{R}} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Linear algebra refresher (4/4)

The nice thing about linear transformations is that we can chain them, i.e. rotation R_1 , followed by rotation R_2 , is equal to rotation R ,

$$\mathbf{R} = \mathbf{R}_2 \cdot \mathbf{R}_1$$

Inverse transformation is performed by inverted matrix, but because R is orthogonal,

$$\mathbf{R}^{-1} = \mathbf{R}^T$$

Nevertheless, and according to transposition rule,

$$\mathbf{R}^{-1} = \mathbf{R}^T = (\mathbf{R}_2 \mathbf{R}_1)^T = \mathbf{R}_1^T \mathbf{R}_2^T$$

Basic rotations

Rotation about X axis

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta \\ 0 & \sin \theta & \cos \theta \end{bmatrix}$$

Rotation about Z axis

$$R_z(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation about Y axis

$$R_y(\theta) = \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix}$$

Rotation + translation

The nice property of homogeneous coordinates and transformations is that we can *treat translation the same way as any other linear transformation, such as rotation.*

$$\tilde{\mathbf{P}} = \tilde{\mathbf{T}}\tilde{\mathbf{R}} = \begin{array}{c} \text{translation} \\ \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array} \begin{array}{c} \text{rotation} \\ \begin{bmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array} = \begin{array}{c} \text{rotation + translation} \\ \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

$$\tilde{\mathbf{P}} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{3 \times 3} & \mathbf{t}_{3 \times 1} \\ \mathbf{0}_{1 \times 3}^T & 1 \end{bmatrix}$$

Remember, \mathbf{R} and \mathbf{t} describe *the pose* of $\{W\}$ w.r.t. $\{C\}$.

Projection matrix

Rotation, translation, projection, discretization

$$w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K}_{3 \times 3} \begin{bmatrix} \mathbf{I}_{3 \times 3} & \mathbf{0}_{3 \times 1} \end{bmatrix} \begin{bmatrix} {}^c X \\ {}^c Y \\ {}^c Z \\ 1 \end{bmatrix} = \mathbf{K}_{3 \times 3} \begin{bmatrix} \mathbf{I}_{3 \times 3} & \mathbf{0}_{3 \times 1} \end{bmatrix} \begin{bmatrix} \mathbf{R}_{3 \times 3} & \mathbf{t} \\ \mathbf{0}_{3 \times 1}^T & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{M}_{3 \times 4} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Internal (intrinsic)
parameters matrix

External (extrinsic)
parameters matrix

Also note that:

$$\mathbf{K}_{3 \times 3} \begin{bmatrix} \mathbf{I}_{3 \times 3} & \mathbf{0}_{3 \times 1} \end{bmatrix} \begin{bmatrix} \mathbf{R}_{3 \times 3} & \mathbf{t} \\ \mathbf{0}_{3 \times 1}^T & 1 \end{bmatrix} = \mathbf{K}_{3 \times 3} \begin{bmatrix} \mathbf{R}_{3 \times 3} & \mathbf{t} \end{bmatrix} = \mathbf{M}_{3 \times 4}$$

Projection matrix

$M = [m_{ij}]$ are numeric parameters of the perspective projection camera model with no explicit physical meaning.

$$w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Compare this equation with eq. for u and v in terms of X, Y, Z and L_i from the previous lectures!

Compare: Direct linear transform (DLT)

- Let's expand

$$u - u_0 = -\frac{f}{\lambda_u} \frac{r_{11}(X - X_0) + r_{12}(Y - Y_0) + r_{13}(Z - Z_0)}{r_{31}(X - X_0) + r_{32}(Y - Y_0) + r_{33}(Z - Z_0)}$$

$$v - v_0 = -\frac{f}{\lambda_v} \frac{r_{21}(X - X_0) + r_{22}(Y - Y_0) + r_{23}(Z - Z_0)}{r_{31}(X - X_0) + r_{32}(Y - Y_0) + r_{33}(Z - Z_0)}$$

- Conventional, more concise notation:

$$u = \frac{L_1 X + L_2 Y + L_3 Z + L_4}{L_9 X + L_{10} Y + L_{11} Z + 1}$$

$$v = \frac{L_5 X + L_6 Y + L_7 Z + L_8}{L_9 X + L_{10} Y + L_{11} Z + 1}$$

- L_i ($i = 1, 2, \dots, 11$) are DLT parameters

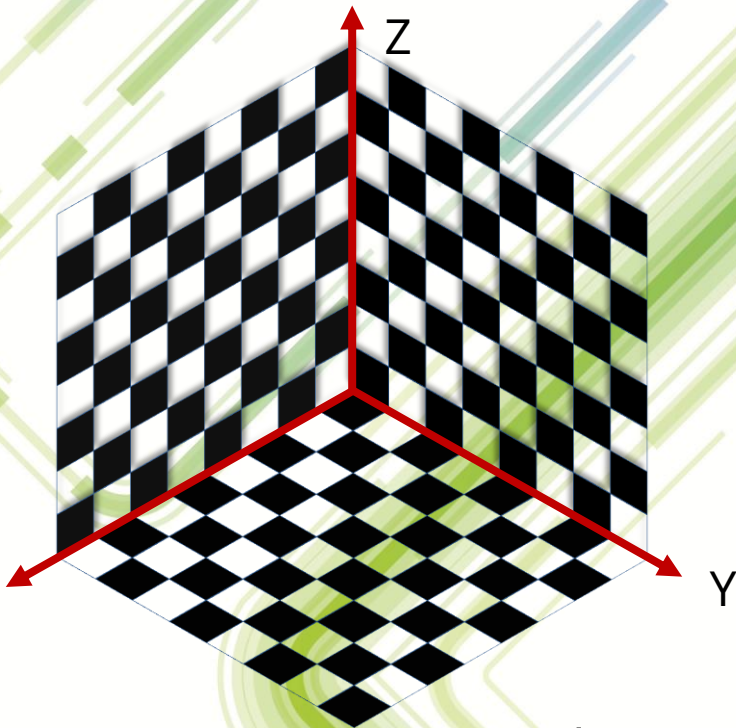
Camera calibration

- Task: derive (unknown) m_{ij} based on a sufficient number of corresponding reference pairs $([u, v]^T, [X, Y, Z]^T)_i$, $(i = 1, 2, 3, \dots)$.

$$w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Camera calibration

Task: derive (unknown) m_{ij} based on a sufficient number of corresponding reference pairs $([u, v]^T, [X, Y, Z]^T)_i$, $(i = 1, 2, 3, \dots, N)$.



To do that a 3D reference object (of known dimensions) is needed.

For each scene point a corresponding image point has to be detected.

At least six, $N \geq 6$, point correspondences are necessary. We *use more* and solve for 11 unknowns in the LS sense.

Note: projection matrix has 12 coefficients, which can be determined only up to a scale. Thus, we set one (m_{34}) coefficient to one and solve for the remaining 11.

Projection matrix decomposition

If \mathbf{M} is known, it is possible to recover physically meaningful camera parameters by matrix decomposition. This can be done, nevertheless, it is numerically quite complex operation.

$$\mathbf{M} = \mathbf{M}_{int} \mathbf{M}_{ext}$$

$$\mathbf{M}_{int} = \begin{bmatrix} f / \lambda_u & \gamma & u_0 \\ 0 & f / \lambda_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{M}_{ext} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}$$

[R | t] matrix inversion

$${}^C\tilde{\mathbf{P}} = \begin{bmatrix} {}^C\mathbf{R}_{3 \times 3} & {}^C\mathbf{t}_{3 \times 1} \\ \mathbf{0}_{3 \times 1}^T & 1 \end{bmatrix}$$

Remember, R and t describe pose {W} w.r.t. {C}.

Homogeneous 4 x 4 matrix, describing *rotation and translation*, has a special structure that makes matrix inversion quite simple.
(this is special case – not true in general for *any* projection matrix!)

$$\tilde{\mathbf{P}}^{-1} = \begin{bmatrix} \mathbf{R}_{3 \times 3} & \mathbf{t}_{3 \times 1} \\ \mathbf{0}_{3 \times 1}^T & 1 \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{R}_{3 \times 3}^T & -\mathbf{R}_{3 \times 3}^T \mathbf{t}_{3 \times 1} \\ \mathbf{0}_{3 \times 1}^T & 1 \end{bmatrix}$$

That simply says that back transformation consists of back rotation \mathbf{R}^T followed by opposite translation with respect to already rotated coordinate frame.

A final note on pose definition

$${}^C\tilde{\mathbf{p}} = \begin{bmatrix} {}^C\mathbf{R}_{3 \times 3} & {}^C\mathbf{t}_{3 \times 1} \\ \mathbf{0}_{3 \times 1}^T & 1 \end{bmatrix}$$

Once again, remember that we have defined \mathbf{R} and \mathbf{t} , i.e. pose of $\{W\}$, with respect to $\{C\}$.

Nevertheless, it is more convenient to define camera pose w.r.t. the world coordinate frame. After all, this is the reference coordinate system.

Thus, let us define pose of $\{C\}$, i.e. \mathbf{R} as well as \mathbf{t} , w.r.t. $\{W\}$.

Therefore, the mapping that takes points w.r.t. the world coordinates frame and maps them to the camera frame, is:

$$\begin{bmatrix} {}^W\mathbf{R}_{3 \times 3}^T & -{}^W\mathbf{R}_{3 \times 3}^T {}^W\mathbf{t}_{3 \times 1} \\ \mathbf{0}_{3 \times 1}^T & 1 \end{bmatrix}$$

With pose of the camera w.r.t. world coordinate system:

$$\begin{bmatrix} {}^W\mathbf{R}_{3 \times 3} & {}^W\mathbf{t}_{3 \times 1} \\ \mathbf{0}_{3 \times 1}^T & 1 \end{bmatrix}$$

A final note on notation

Sometimes, when many coordinate frames are involved.
e.g. world, two or more robots, two or more cameras with their own coordinate frames, it is useful to explicitly denote the relations and transformations among them.

${}^B\mathbf{T}_A$

T is a mapping of A to B, where T is a 'pose' of coordinate frame A with respect to B.

${}^A\tilde{\mathbf{p}}$

P is a vector (homogeneous vector) defined w.r.t. coordinate frame A.

For instance, composition of transformations that maps p from A to B, then from B to C, from C to D, reads:

$${}^D\mathbf{p} = {}^D\mathbf{T}_C {}^C\mathbf{T}_B {}^B\mathbf{T}_A {}^A\mathbf{p} = {}^D\mathbf{T}_A {}^A\mathbf{p}$$

A simple example (1/5)

- Let us take the camera:
 - with 1/2" sensor (8 mm diagonal, aspect ratio $W/H = 4/3$)
 - $W \times H = 6,4\text{mm} \times 4,8\text{ mm} = 1280 \times 960$ pixels
 - Pixel size = 5 μm (microns), square pixels
 - Principal point is in the center of the sensor, no skew, no lens distortion.
 - Camera lens has $f = 8\text{ mm}$
- Task: Derive camera calibration matrix \mathbf{K}
 - Remember: \mathbf{K} is a matrix of *internal* camera parameters

$$\mathbf{K} = \begin{bmatrix} f_u & \gamma & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1600 & 0 & 640 \\ 0 & 1600 & 480 \\ 0 & 0 & 1 \end{bmatrix}$$

A simple example (2/5)

Let's imagine a single scene point ${}^C P = [0.25, 0.10, 0.75]^T$ (in meters) w.r.t. $\{C\}$.

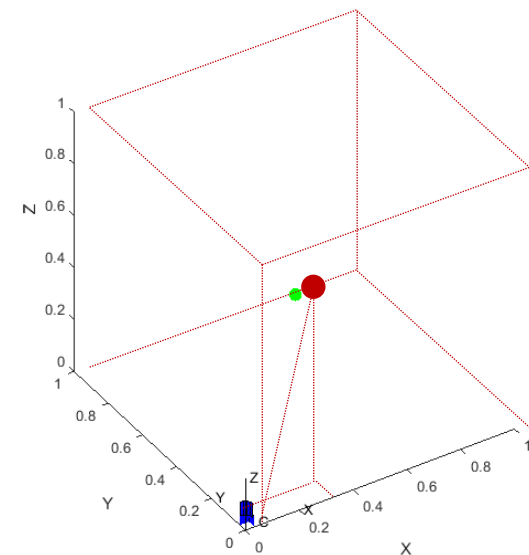
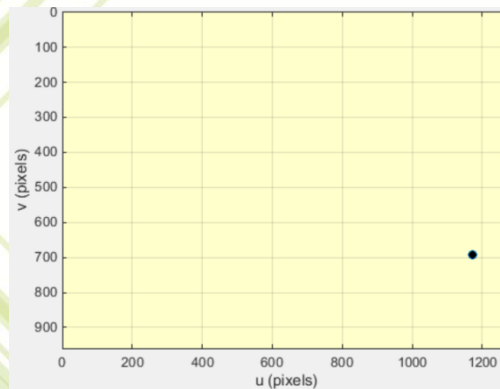
Also, $\{C\}$ coincides with $\{W\}$.

The homogeneous coordinates of point in the image are:

$$\tilde{p} = \mathbf{K} \begin{bmatrix} {}^C X \\ {}^C Y \\ {}^C Z \end{bmatrix} = \begin{bmatrix} 1600 & 0 & 640 \\ 0 & 1600 & 480 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.25 \\ 0.10 \\ 0.75 \end{bmatrix} = \begin{bmatrix} 880 \\ 520 \\ 0.75 \end{bmatrix} = 0.75 \begin{bmatrix} 880/0.75 \\ 520/0.75 \\ 1 \end{bmatrix} = 0.75 \begin{bmatrix} 1173 \\ 693 \\ 1 \end{bmatrix}$$

The point position in the image is:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 1173 \\ 693 \end{bmatrix}$$



A simple example (3/5)

Now, let's move the camera w.r.t. {W} for $t^T = [0.50 \ 0.20 \ 1.00]^T$

And rotate the camera for 180 deg. around X axis.

Therefore, poses of the camera w.r.t. {W}, and of {W} w.r.t. camera are:

$${}^W\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0.50 \\ 0 & -1 & 0 & 0.20 \\ 0 & 0 & -1 & 1.00 \\ 0 & 0 & 0 & 1 \end{bmatrix} \Rightarrow {}^C\mathbf{P} = {}^W\mathbf{P}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0.50 \\ 0 & -1 & 0 & 0.20 \\ 0 & 0 & -1 & 1.00 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 & 0 & -0.50 \\ 0 & -1 & 0 & 0.20 \\ 0 & 0 & -1 & 1.00 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Let's take a point $P = [0.25 \ 0.1 \ 0.0]^T$ w.r.t. {W}. Its coordinates w.r.t. {C} are:

$${}^C\tilde{P} = \begin{bmatrix} {}^CX \\ {}^CY \\ {}^CZ \\ 1 \end{bmatrix} = {}^C\mathbf{P} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & -0.50 \\ 0 & -1 & 0 & 0.20 \\ 0 & 0 & -1 & 1.00 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.25 \\ 0.10 \\ 0.00 \\ 1 \end{bmatrix} = \begin{bmatrix} -0.25 \\ 0.10 \\ 1.00 \\ 1 \end{bmatrix}$$

A simple example (4/5)

The projection matrix is,

$$\mathbf{M} = \begin{bmatrix} 1600 & 0 & 640 \\ 0 & 1600 & 480 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -0.50 \\ 0 & -1 & 0 & 0.20 \\ 0 & 0 & -1 & 1.00 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1600 & 0 & -640 & -160 \\ 0 & -1600 & -480 & 800 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

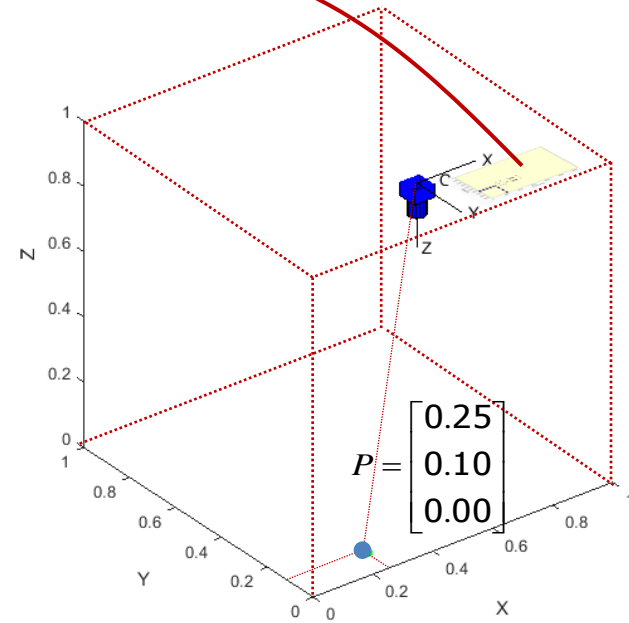
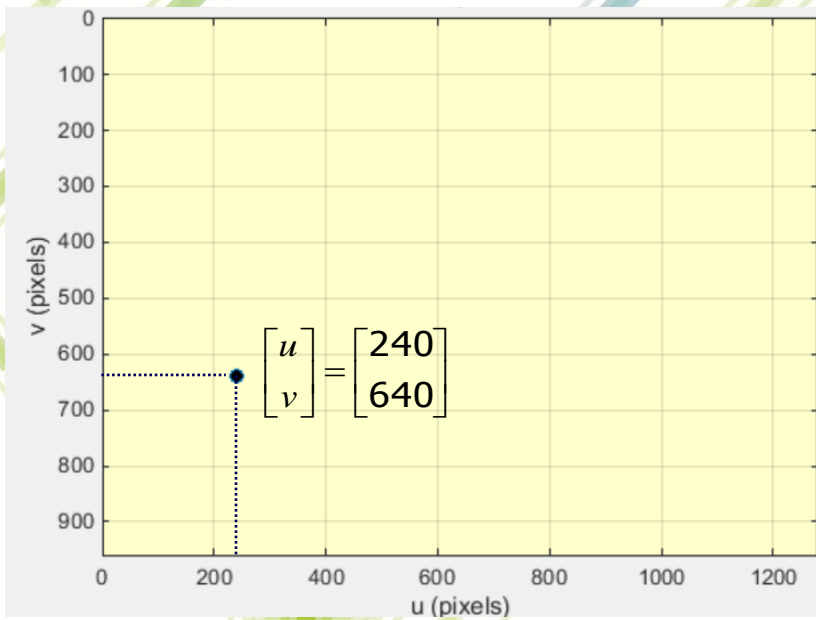
And the corresponding image point is:

$$w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} 1600 & 0 & -640 & -160 \\ 0 & -1600 & -480 & 800 \\ 0 & 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} 0.25 \\ 0.10 \\ 0.00 \\ 1 \end{bmatrix} = \begin{bmatrix} 240 \\ 640 \\ 1 \end{bmatrix}$$

You should get the same result from

$$\tilde{p} = \mathbf{K} \begin{bmatrix} {}^cX \\ {}^cY \\ {}^cZ \end{bmatrix} = \begin{bmatrix} 1600 & 0 & 640 \\ 0 & 1600 & 480 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -0.25 \\ 0.10 \\ 1.00 \end{bmatrix} \quad \text{Try it!}$$

A simple example (5/5)



Plane to plane mapping

$$Z=0$$

$$w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix}$$

$$w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K} [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t}] \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

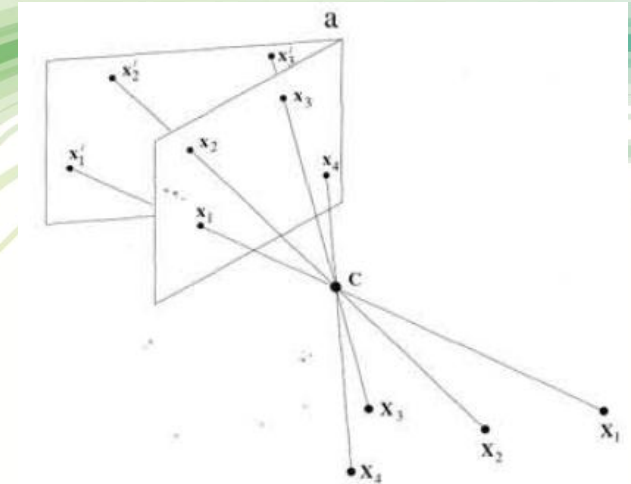
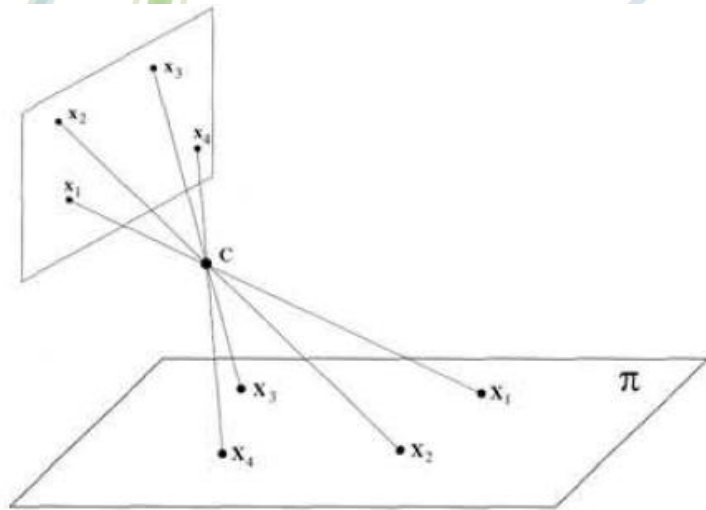
H - homography (projective transformation)
3 x 3 matrix, full rank, invertible, 8 DoF

$$\mathbf{H}_{3 \times 3} = \mathbf{K} [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t}]$$

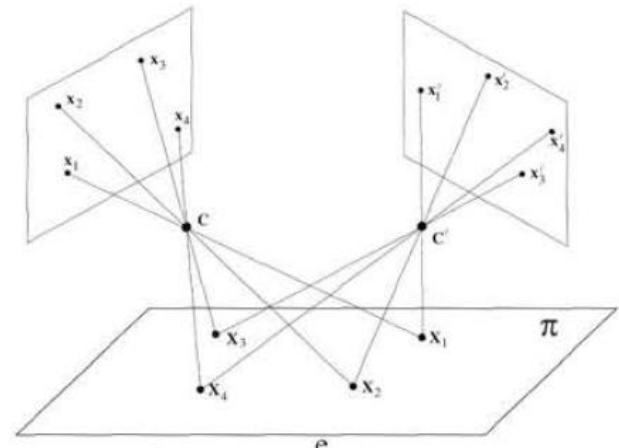
$$w\mathbf{u} = \mathbf{H}\mathbf{p}$$

Each point pair contributes 2 equations, thus 4 points are sufficient to solve for H.

Plane to plane mapping



R. Hartley, A. Zisserman
Multiple View Geometry in Computer Vision, Second Edition, 2003



Camera calibration toolbox

- A very popular and powerful toolbox for camera calibration written by *Jean-Yves Bouguet*.
- Works in Matlab
- His implementation is largely based on the Z. Zhang paper.
- It uses some (possibly large) number of 2D calibration patterns instead of a 3D calibration object.
- It also does lens distortion correction, stereo calibration, and more.
- It is also well documented
- https://www.vision.caltech.edu/bouguetj/calib_doc/

Further reading

E. Trucco, A. Verri, *Introductory Techniques for 3D Computer Vision*, Prentice Hall, 1998.

D. Forsyth, J. Ponce, *Computer Vision: a modern approach*, Pearson 2012, Part I, Chapter 2.

P. Corke, *RVC book*, Springer 2013, Part IV.

M. Sonka, V. Hlaváč, R. Boyle, *Image Processing, Analysis, and Machine Vision*, Thomson, 2008.

Z. Zhang, *A flexible new technique for camera calibration*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(11):1330–1334, 2000.

https://www.vision.caltech.edu/bouguetj/calib_doc/

The background features a series of overlapping, curved lines in various shades of green and blue, creating a sense of depth and movement. The lines are of varying thicknesses and some have a slight gradient, giving them a three-dimensional appearance. They are arranged in a way that suggests a complex, interconnected network or a stylized representation of a landscape or architectural structure.

Questions?