Image formation: Projective geometry

Peter Veelaert, Hiep Luong

February 12, 2024

Table of Contents

Table of Contents

2D projections from 3D scenes

- How can we represent points at infinity?
- Is there a mathematical framework in which parallelism and concurrency are instances of the same concept?

Homogeneous coordinates

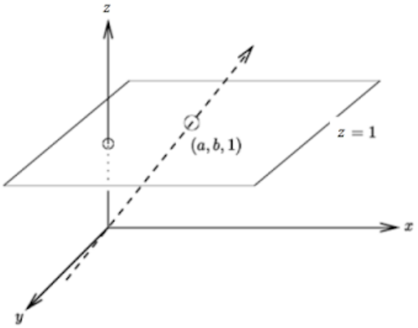
- Homogeneous coordinates are better suited to describe projections than Cartesian coordinates
- used in projective geometry (geometry of projections)
- aka projective coordinates
- real projective plane = Euclidean plane + points at infinity

The projective plane

- 2D point is represented by triple (X, Y, Z), where X, Y, Z are not all zero
- (X, Y, Z) and $(\lambda X, \lambda Y, \lambda Z)$ represent the same point for all $\lambda \neq 0$
- when $Z \neq 0$, (X, Y, Z) in the projective plane corresponds to $(X/Z, Y/Z)$ in the Euclidean plane
- $(X, Y, 0)$ is a point at infinity in the projective plane (Euclidean plane has no such points)
- the points $(X, Y, 0)$ lie at the line at infinity

Projective vs Cartesian coordinates

Projective	Cartesian
$(4, 3, 1)$	$(4, 3)$
$(8, 6, 2)$	$(4, 3)$
$(-8, -6, -2)$	$(4, 3)$
$(4\lambda, 3\lambda, \lambda)$	$(4, 3)$
$(4, 3, 0)$	non existent
$(\lambda X, \lambda Y, \lambda)$	(X, Y)



- All $(x, y, z) \neq (0, 0, 0)$ on the line passing through $(a, b, 1)$ represent the same projective point
- lines in the xy-plane correspond to points at infinity

Lines in the projective plane

Equation of line:

$$ax + by + cz = 0$$

- a, b, c are constants
- equation is homogeneous (all terms have same degree)
- (a, b, c) and $\lambda(a, b, c)$ (where $\lambda \neq 0$) represent the same line
- line passes through the point $(b, -a, 0)$ at infinity
- each line (not at infinity) meets the line at infinity in exactly one point

Lines in the projective plane

Projective	Cartesian
$ax + by + cz = 0$	$ax + by + c = 0$

When (X, Y, Z) in the projective plane lies on a line with coefficients (a, b, c),
then $(X/Z, Y/Z)$ in the Euclidean plane lies on a line defined by the same coefficients (a, b, c).

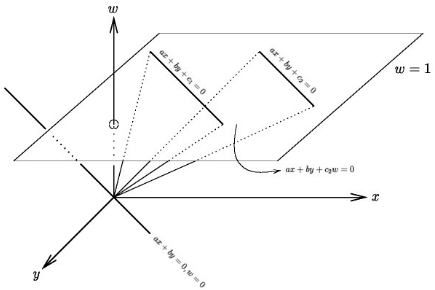
Lines in the projective plane

Parallel lines in the Euclidean plane,

$$ax+by+c_1 = 0, \quad ax+by+c_2 = 0$$

correspond to lines in the projective plane that meet at a common point $(b, -a, 0)$ at infinity.

Thus parallelism is replaced by concurrency at infinity.



- Each line in the affine plane $w = 1$ corresponds to a plane in xyw-space
- the two lines in the affine plane meet at infinity at the point represented by $ax + by = 0, w = 0$

Lines and points

- The point (X, Y, Z) lies on the line defined by (a, b, c) if $aX + bY + cZ = 0$
- This can be rewritten in vector notation as dot product:

$$0 = \begin{bmatrix} a & b & c \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \mathbf{l}^T \mathbf{p}$$

- A line is also represented as a homogeneous 3-vector **l**.

Calculations with lines and points

- A line can be defined by two points $\mathbf{x}$ and $\mathbf{y}$ (cross product of the vectors, think of the right hand rule):

$$\mathbf{l} = \mathbf{x} \times \mathbf{y}$$

- Proof: Let the line be constructed as follows

$$\mathbf{l} \triangleq \mathbf{x} \times \mathbf{y},$$

  we then can proof that points $\mathbf{x}$ and $\mathbf{y}$ are lying on this line:

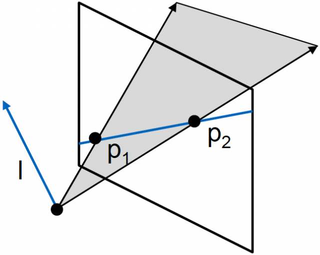
$$\mathbf{l}^T\mathbf{x} = \mathbf{x}^T\mathbf{l} = \mathbf{x}^T(\mathbf{x} \times \mathbf{y}) = 0, \mathbf{l}^T\mathbf{y} = \mathbf{y}^T\mathbf{l} = \mathbf{y}^T(\mathbf{x} \times \mathbf{y}) = 0$$

  (indeed, the scalar triple product is zero because the rank of the matrix spanned by the three vectors is lower than 3)
- Analogously, a point can be defined as the intersection of two lines $\mathbf{l}$ and $\mathbf{m}$:

$$\mathbf{x} = \mathbf{l} \times \mathbf{m}$$

Geometric interpretation of line parameters

- A line $\mathbf{l}$ is a homogeneous 3-vector, which is a ray in projective space
- It is $\perp$ to every point ray $\mathbf{p}$ on the line: $\mathbf{l}^T\mathbf{p} = 0$



- $\mathbf{l}$ is $\perp$ to $\mathbf{p}_1$ and $\mathbf{p}_2 \Rightarrow \mathbf{l} = \mathbf{p}_1 \times \mathbf{p}_2$
- $\mathbf{l}$ is the plane normal

Point and line duality

- Duality principle: to any theorem of 2-dimensional projective geometry there corresponds a dual theorem, which may be derived by interchanging the role of points and lines in the original theorem.
- Some examples:

$$\mathbf{p} \longleftrightarrow \mathbf{l}$$

$$\mathbf{p}^T\mathbf{l} = 0 \longleftrightarrow \mathbf{l}^T\mathbf{p} = 0$$

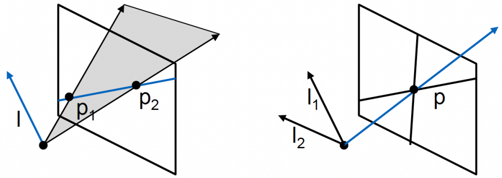
$$\mathbf{p} = \mathbf{l} \times \mathbf{l}' \longleftrightarrow \mathbf{l} = \mathbf{p} \times \mathbf{p}'$$

$$\dots$$

Geometric interpretation of line intersection

- Line parameters defined by two points:

$$\mathbf{l} \text{ is } \perp \text{ to } \mathbf{p}_1 \text{ and } \mathbf{p}_2 \Rightarrow \mathbf{l} = \mathbf{p}_1 \times \mathbf{p}_2$$



- Point parameters by intersection of two lines:

$$\mathbf{p} \text{ is } \perp \text{ to } \mathbf{l}_1 \text{ and } \mathbf{l}_2 \Rightarrow \mathbf{p} = \mathbf{l}_1 \times \mathbf{l}_2$$

Intersection of parallel lines

- Suppose that **l** and **m** are two parallel lines:

$$\mathbf{l} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}, \mathbf{m} = \begin{bmatrix} a \\ b \\ d \end{bmatrix}$$

- Intersection of **l** and **m** is given by

$$\mathbf{p} = \mathbf{l} \times \mathbf{m} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \times \begin{bmatrix} a \\ b \\ d \end{bmatrix} = \begin{bmatrix} bd - bc \\ ac - ad \\ ab - ab \end{bmatrix} = (d - c) \begin{bmatrix} b \\ -a \\ 0 \end{bmatrix}$$

Projective space

Projective	Cartesian
(X, Y, Z, W)	$(X/W, Y/W, Z/W)$

- $(X, Y, Z, 0)$ lies on the plane at infinity
- each line (not at infinity) meets the plane at infinity in exactly one point
- each plane meets the plane at infinity in a line at infinity

Projective geometry: summary

- Projective geometry extends ordinary geometry with *ideal* points/lines/planes - where parallel lines/planes meet!
- Ideal points/lines/planes lie at infinity.
- 1D: projective line = ordinary line + ideal point
- 2D: projective plane = ordinary plane + ideal line
- 3D: projective space = ordinary space + ideal plane
- 2D: two parallel lines intersect in an ideal point
- 3D: two parallel planes intersect in an ideal line
- 2D: point and line duality
- 3D: point and plane duality (not lines!)

Table of Contents

Hierarchy of 2D transformations

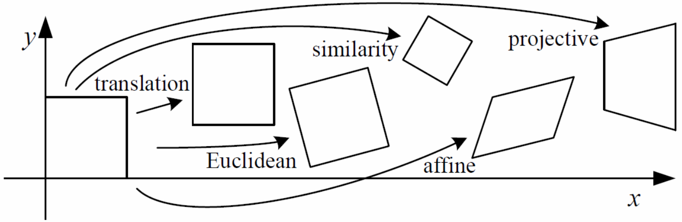


Figure from Szeliski, Computer Vision: Algorithms and Applications

Hierarchy of 2D transformations

transformation	matrix	dof	invariants
			$\hookrightarrow$ degree of freedom
Euclidean	$\begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$	3	lengths, areas, ...
Similarity	$\begin{bmatrix} sr_{11} & sr_{12} & t_x \\ sr_{21} & sr_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$	4	angles,...
Affine	$\begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$	6	parallelism, ratio areas,...
Projective	$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$	8	collinearity, concurrency, ...

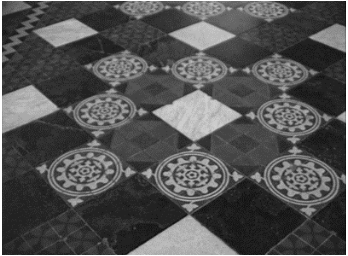
Similarity transformation

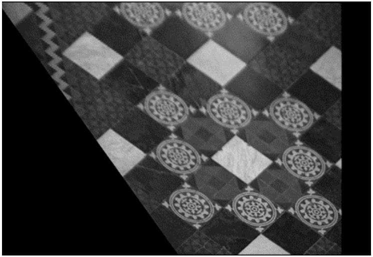
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

preserves angles, length ratios
four parameters:

- scaling parameter s
- rotation angle θ
- translation t_x, t_y

Affine transformation

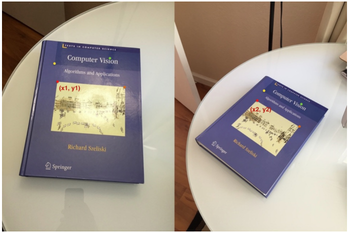




$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & t_x \\ c & d & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- allows skew, preserves parallelism
- six parameters

2D homography



$$\begin{bmatrix} \lambda x' \\ \lambda y' \\ \lambda \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- can map image of planar object A onto any other image of A
- points at infinity can be mapped onto image points
- 9 parameters, but only 8 degrees of freedom (arbitrary scaling)

Table of Contents

Hierarchy of 3D transformations

Transformation	Matrix	# DoF	Preserves	Icon
translation	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{3\times 4}$	3	orientation	
	+3 rotation ([illegible])			
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{3\times 4}$	6	lengths	
similarity	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{3\times 4}$ (+ scaling)	7	angles	
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{3\times 4}$	12	parallelism	
projective	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}_{4\times 4}$	15	straight lines	

Table 2.2 from Szeliski, Computer Vision: Algorithms and Applications

Hierarchy of 3D transformations

Note that, depending on the application, we can either use 3 × 3, 3 × 4, or 4 × 4 matrices to represent 3D transformations.

If we compose rigid, similarity, affine or projective transforms, we must use 4 × 4 matrices. For example,

$$\mathbf{A}_i = \underbrace{\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\text{4x4 moet iteraties}}, \quad \mathbf{A} = \mathbf{A}_1\mathbf{A}_2\mathbf{A}_2, \quad \mathbf{P} = \mathbf{H}_1\mathbf{A}_1, \ldots$$

Hetzelfde werkt enkel bij compositie

If we apply a single non-projective transformation, we may discard the fourth row:

3x4

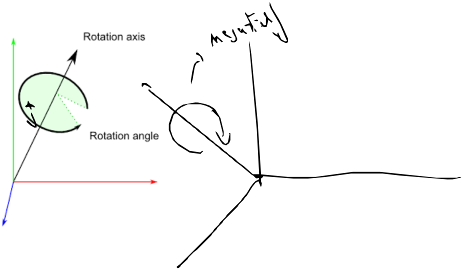
$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

For pure rotations without translation, we can use 3 × 3 matrices, For example,

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}, \quad \mathbf{R} = \mathbf{R}_1\mathbf{R}_2\mathbf{R}_3$$

Voor alle kan je 4x4 gebruiken

3D rotations



- any rotation in 3D is a rotation about an axis $\hat{\mathbf{n}}$
- any composition of rotations about different axes can be replaced by one rotation about one axis
- rotations in 3D are not commutative

3D rotations: cross products

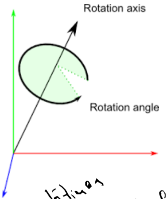
For a given vector $(\hat{n}_x, \hat{n}_y, \hat{n}_z)$, define the anti-symmetric matrix

$$[\hat{\mathbf{n}}]_\times = \begin{bmatrix} 0 & -\hat{n}_z & \hat{n}_y \\ \hat{n}_z & 0 & -\hat{n}_x \\ -\hat{n}_y & \hat{n}_x & 0 \end{bmatrix}$$

Then any cross product $\hat{\mathbf{n}} \times \mathbf{v}$ can be written as

$$\hat{\mathbf{n}} \times \mathbf{v} = [\hat{\mathbf{n}}]_\times \mathbf{v}.$$

3D rotations: Rodriguez's formula



↪ rotatieas ↪ hoek

Rotation matrix about axis $\hat{\mathbf{n}}$ over θ:

Handig om rotatie matrix op te stellen

!

$$\mathbf{R}(\hat{\mathbf{n}}, \theta) = \mathbf{I} + \sin\theta\,[\hat{\mathbf{n}}]_\times + (1 - \cos\theta)\,[\hat{\mathbf{n}}]_\times^2$$

where $\mathbf{I}$ is identity matrix. This is known as Rodriguez's formula.

3D rotations

Example

Rotation matrix about axis $\hat{\mathbf{n}} = (0, 0, 1)$ over θ:

$$[\hat{\mathbf{n}}]_\times = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$[\hat{\mathbf{n}}]_\times^2 = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{R}(\hat{\mathbf{n}}, \theta) = \mathbf{I} + \sin\theta\,[\hat{\mathbf{n}}]_\times + (1 - \cos\theta)\,[\hat{\mathbf{n}}]_\times^2 =$$

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

Camera frames vs world frames

- For the pinhole camera model we will need to convert world coordinates into camera coordinates
- Any change of coordinate frame can be decomposed into a rotation + translation

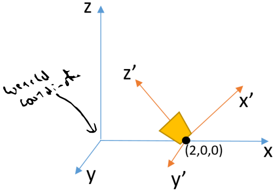
In particular the transformation of world coordinates $(X_W, Y_W, Z_W, 1)$ into camera coordinates $(X_C, Y_C, Z_C, 1)$ can be written as

$$\begin{pmatrix} X_C \\ Y_C \\ Z_C \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R} & \mathbf{T} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{pmatrix}$$

with

- rotation matrix $\mathbf{R}$
- translation vector $\mathbf{T}$

Camera coordinate frame: an example



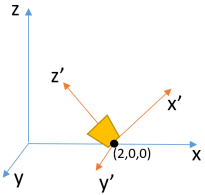
A camera is located at position (2,0,0) and rotated over 45 degrees about the y-axis. Find the change-of-basis matrix that maps world coordinates (x, y, z) onto camera coordinates (x', y', z').

Suppose the camera is first located at the origin and that its optical axis is aligned with the z-axis. Then the camera moves to $(2, 0, 0)$ and rotates about the y-axis. This transformation can be described by

translation

$$\mathbf{W} = \begin{pmatrix} \sqrt{2}/2 & 0 & -\sqrt{2}/2 & 2 \\ 0 & 1 & 0 & 0 \\ \sqrt{2}/2 & 0 & \sqrt{2}/2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Camera coordinate: an example



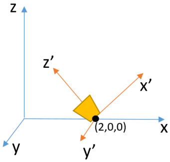
For example, when the camera moves, the point $(0, 0, 1)$ on the optical axis of the camera moves to its new position

camera => world

world coordinates

$$\begin{pmatrix} \sqrt{2}/2 & 0 & -\sqrt{2}/2 & 2 \\ 0 & 1 & 0 & 0 \\ \sqrt{2}/2 & 0 & \sqrt{2}/2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 2-\sqrt{2}/2 \\ 0 \\ \sqrt{2}/2 \\ 1 \end{pmatrix} \approx \begin{pmatrix} 1.293 \\ 0 \\ 0.707 \\ 1 \end{pmatrix}$$

Camera coordinate: an example

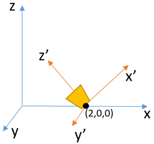


If the basis vectors $\mathbf{e}_i$ of a coordinate frame transform as $\mathbf{W}\mathbf{e}_i$, the coordinates of a vector $\mathbf{v} = (v_1, v_2, v_3, 1)^T$ transform as $\mathbf{W}^{-1}\mathbf{v}$, where

$$\mathbf{W}^{-1} = \begin{pmatrix} \sqrt{2}/2 & 0 & \sqrt{2}/2 & -\sqrt{2} \\ 0 & 1 & 0 & 0 \\ -\sqrt{2}/2 & 0 & \sqrt{2}/2 & \sqrt{2} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

world → camera.

Camera coordinate: an example



For example, the **camera coordinates** of the origin of the world coordinate system are

origin

$$\begin{pmatrix} \sqrt{2}/2 & 0 & \sqrt{2}/2 & -\sqrt{2} \\ 0 & 1 & 0 & 0 \\ -\sqrt{2}/2 & 0 & \sqrt{2}/2 & \sqrt{2} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} -\sqrt{2} \\ 0 \\ \sqrt{2} \\ 1 \end{pmatrix} \approx \begin{pmatrix} -1.414 \\ 0 \\ 1.414 \\ 1 \end{pmatrix}$$

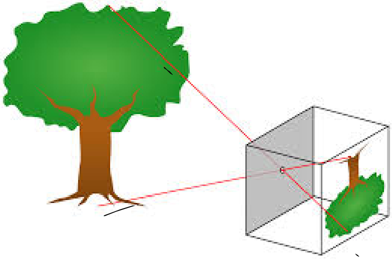
Likewise, the **camera coordinates** of the point with world coordinates $(0, 0, 1, 1)$ are

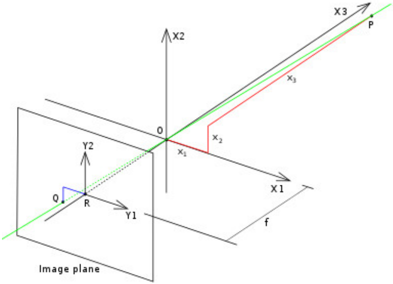
$$\begin{pmatrix} \sqrt{2}/2 & 0 & \sqrt{2}/2 & -\sqrt{2} \\ 0 & 1 & 0 & 0 \\ -\sqrt{2}/2 & 0 & \sqrt{2}/2 & \sqrt{2} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} -\sqrt{2}/2 \\ 0 \\ 3\sqrt{2}/2 \\ 1 \end{pmatrix} \approx \begin{pmatrix} -0.707 \\ 0 \\ 2.12 \\ 1 \end{pmatrix}$$

Table of Contents

Pinhole model

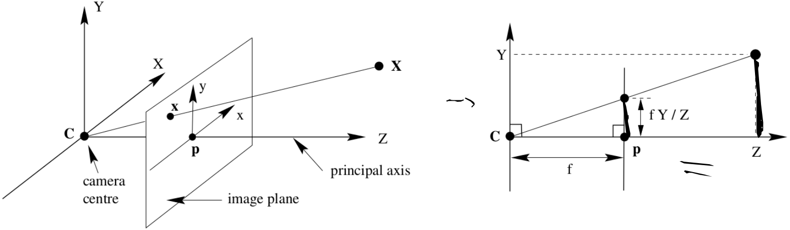
camera obscura = box with a small hole





But we may put the image plane as well between the projection center and the object...

Pinhole model



The image point (x, y) is found by triangulation:

$$x = f\frac{X}{Z}, \quad y = f\frac{Y}{Z}.$$

$\Leftrightarrow \quad \frac{x}{X} = \frac{f}{Z} \qquad \frac{y}{Y} = \frac{f}{Z}$

Projection with homogeneous coordinates

In homogeneous coordinates, these relations can be rewritten as:

$$Z \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

which is equivalent to

$$\begin{aligned} Zx &= fX \\ Zy &= fY \\ Z1 &= 1Z \end{aligned}$$

or

$$x = f\frac{X}{Z}, \quad y = f\frac{Y}{Z}$$

Projection with homogeneous coordinates

Letting

$$K_f = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

the projection has the matrix representation

$$Z \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = K_f \mathbf{X}$$

or

$$\lambda \mathbf{x} = K_f \mathbf{X}$$

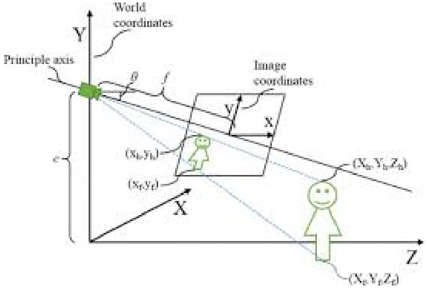
where at the left side λ need not be known.

By introducing homogeneous coordinates, the division by Z disappears, and the projection takes the form of a **linear transformation** in the vector space of homogeneous coordinates.

Camera coordinates vs world coordinates

The general case: camera axis not aligned with z-axis.

- We use two separate coordinate systems: a **world coordinate system** and a camera coordinate system attached to the camera, with the Z_c axis aligned with the optical axis.
- **World coordinates $\mathbf{X}_W$** are first transformed into **camera coordinates $\mathbf{X}_C$**.
- Once we have the camera coordinates, the projection takes the simple form $\lambda \mathbf{x} = K_f \mathbf{X}_C$



$$\begin{pmatrix} X_C \\ Y_C \\ Z_C \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R} & \mathbf{T} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{pmatrix}$$

with

- rotation matrix $\mathbf{R}$
- translation vector $\mathbf{T}$

elements of $\mathbf{R}$ and $\mathbf{T}$ are called the **extrinsic parameters**.

Image coordinates vs pixel coordinates

Furthermore, in the image plane the position of an image point is usually expressed by **pixel coordinates** instead of **image coordinates**:

→ upper triangular matrix

→ intrinsic (Bildkoordinaten)

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} s_x & s_\theta & x_0 \\ 0 & s_y & y_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X_C \\ Y_C \\ Z_C \\ 1 \end{pmatrix}$$

$s_x, s_y, s_\theta, x_0, y_0$ are called **intrinsic camera parameters**.

Important remark. To find the real positions of all objects and points, one must express all coordinates with the same units, e.g. millimeters, pixelsize, ...

in the plane ?

Camera model

Combining all transformations:

- world coordinates → camera coordinates
- projection onto the image plane
- image plane coordinates → pixel coordinates

intrinsic $\qquad$ 3×4 $\qquad$ extrinsic

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} s_x & s_\theta & x_0 \\ 0 & s_y & y_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{R} & \mathbf{T} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{pmatrix}$$

Or

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \mathbf{K} \left(\mathbf{I} | \mathbf{0} \right) \begin{pmatrix} \mathbf{R} & \mathbf{T} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{pmatrix}$$

Or

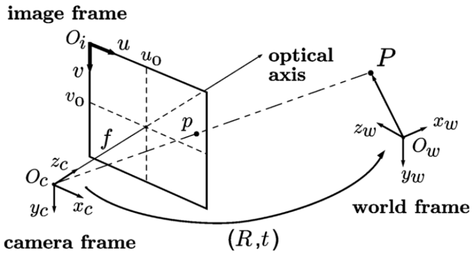
world → camera.

$$\underline{\lambda \mathbf{x} = \mathbf{P} \mathbf{X}}$$

where **P** is the 3 × 4 **projection matrix**.

Projective cameras

Final result. The projection of a pinhole camera can be represented by a single 3×4 matrix



$(X, Y, Z, 1)^T$ is projected onto image point $(x, y, 1)^T$ by

$$\begin{bmatrix} \lambda x \\ \lambda y \\ \lambda \end{bmatrix} = \mathbf{PX} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

11 dof.

non special

Projective cameras

Summary. The projection matrix for a camera with a spherical lens can be decomposed as:

$$\begin{bmatrix} \lambda x \\ \lambda y \\ \lambda \end{bmatrix} = \mathbf{P} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{K} \left[\mathbf{R} | \mathbf{t} \right] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \quad \text{with} \quad \mathbf{K} = \begin{bmatrix} f & s & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

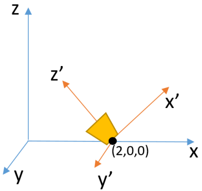
where

- **K** is the **camera calibration matrix**, which contains the **intrinsic** parameters

  principiaal as = z-as.
- f is the focal length
- s is skew; often $s \approx 0$
- (p_x, p_y) coordinates of principal point; often $(p_x, p_y) \approx (0, 0)$
- **R** is a 3x3 rotation matrix,
- $\mathbf{t} = (t_x, t_y, t_z)^T$ represents a translation
- **R**, **t** contain the **extrinsic** camera parameters
- **P** has 3 + 3 + 4 dof (3 + 3 extrinsic, 4 intrinsic)

  → enkel voor sferische lenzen
- for non spherical (e.g., cylindrical) lenses we must replace f by f_x, f_y (5 intrinsic parameters)

Projective camera: an example



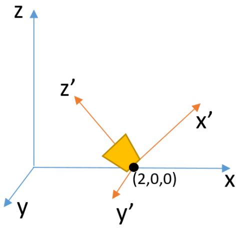
Suppose our camera is located at position (2,0,0) and rotated over 45 degrees about the *y*-axis, and that the camera calibration matrix is

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

then

$$\mathbf{P} = \begin{bmatrix} \sqrt{2}/2 & 0 & \sqrt{2}/2 & -\sqrt{2} \\ 0 & 1 & 0 & 0 \\ -\sqrt{2}/2 & 0 & \sqrt{2}/2 & \sqrt{2} \end{bmatrix}$$

Projective camera: an example



For example, the point with world coordinates (0,0,2,1) is projected onto

$$\begin{bmatrix} \sqrt{2}/2 & 0 & \sqrt{2}/2 & -\sqrt{2} \\ 0 & 1 & 0 & 0 \\ -\sqrt{2}/2 & 0 & \sqrt{2}/2 & \sqrt{2} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 2\sqrt{2} \end{bmatrix} = 2\sqrt{2} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

which is at the center of the image.

Table of Contents

Projection of planar objects

On a planar object, all points satisfy $Z = aX + bY + c$, for some coefficients a, b, c.

vech!

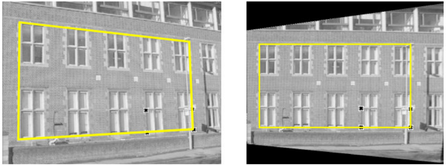
We can eliminate Z from the projection, by absorbing the relation between X, Y, Z in a 3×3 projection matrix.

3 x 3 ronde 2

$$\begin{bmatrix} \lambda x \\ \lambda y \\ \lambda \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ aX + bY + c \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} + ap_{13} & p_{12} + bp_{13} & p_{14} + cp_{13} \\ p_{21} + ap_{23} & p_{22} + bp_{23} & p_{24} + cp_{23} \\ p_{31} + ap_{33} & p_{32} + bp_{33} & p_{34} + cp_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

Hence, the projection of a planar object can be represented by a 2D homography.

2D Homographies



from Hartley & Zisserman

Similarly, two image projections of a **planar object** are related by a homography (= transformation of the 2D projective plane):

$$\lambda \begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix}$$

In fact, if $\lambda_1 \mathbf{x}_1 = \mathbf{H}_1 \mathbf{X}$ and $\lambda_2 \mathbf{x}_2 = \mathbf{H}_2 \mathbf{X}$,
then $\lambda_1 \mathbf{H}_1^{-1} \mathbf{x}_1 = \lambda_2 \mathbf{H}_2^{-1} \mathbf{x}_2$,
hence $\lambda_1 \mathbf{H}_2 \mathbf{H}_1^{-1} \mathbf{x}_1 / \lambda_2 = \mathbf{x}_2$ or $\mathbf{H}\mathbf{x}_1 = \lambda \mathbf{x}_2$

2D homography

How can we find this homography?

Let $(x, y, 1)^T$ and $\lambda(x', y', 1)^T$ be a pair of corresponding points.

$$\lambda \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Then

$$\begin{aligned} \lambda x' &= h_{11}x + h_{12}y + h_{13} \\ \lambda y' &= h_{21}x + h_{22}y + h_{23} \\ \lambda &= h_{31}x + h_{32}y + h_{33} \end{aligned}$$

Or,

$$\begin{aligned} x'(h_{31}x + h_{32}y + h_{33}) &= h_{11}x + h_{12}y + h_{13} \\ y'(h_{31}x + h_{32}y + h_{33}) &= h_{21}x + h_{22}y + h_{23} \end{aligned}$$

2D homography

$$\begin{aligned} x'(h_{31}x + h_{32}y + h_{33}) &= h_{11}x + h_{12}y + h_{13} \\ y'(h_{31}x + h_{32}y + h_{33}) &= h_{21}x + h_{22}y + h_{23} \end{aligned}$$

Define

$$\mathbf{h} = (h_{11}\ h_{12}\ h_{13}\ h_{21}\ h_{22}\ h_{23}\ h_{31}\ h_{32}\ h_{33})$$

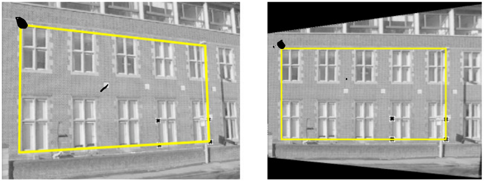
and

$$\begin{aligned} \mathbf{v}_1 &= (x\ y\ 1\ 0\ 0\ 0\ -xx'\ -yx'\ -x') \\ \mathbf{v}_2 &= (0\ 0\ 0\ x\ y\ 1\ -xy'\ -yy'\ -y') \end{aligned}$$

Then this pair provides two constraints:

$$\mathbf{v}_1^T\mathbf{h} = 0, \quad \mathbf{v}_2^T\mathbf{h} = 0$$

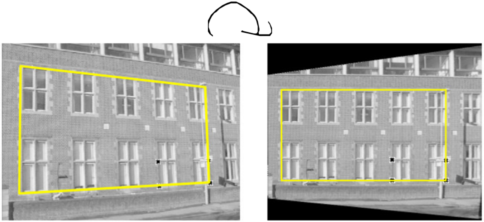
2D Homographies



from Hartley & Zisserman

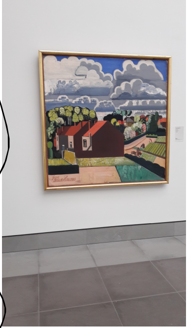
- by matching the corners of the two quadrangles we obtain 4 pairs
- 4 point pairs provide 8 constraints
- 8 constraints are sufficient to determine the homography matrix which has 8 degrees of freedom, (e.g. let $h_{33} = 1$)
- to get higher accuracy: use N points, $N > 4$, and Singular Value Decomposition (SVD) to find a Least Squares (LS) solution of a system of $2N + 1$ equations in 9 unknowns

2D Homographies



- homographies are often used to correct the perspective deformation of a flat scene

2D Homographies



2 vlakken → 2
2 2D homografie

- A painting is a planar object.
- Hence, the painting on the left can be mapped onto the painting on the right by a 2D homography.
- To map the floor tiles we need another 2D homography.

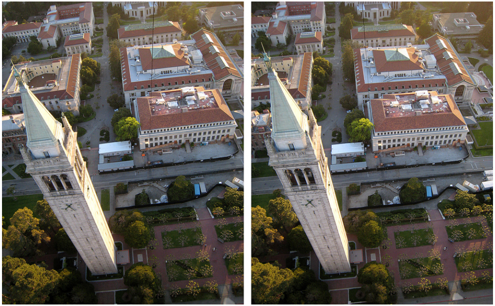
2D Homographies

Can we find this homography without indicating point pairs manually?

- use feature detector/descriptor to find N (N $>>$ 4) point pairs (see later)
- use Ransac (or MLESAC) to find a set of 4 good point pairs (see later)
- this technique works well and is often employed

2D homographies vs 3D projections

hier ga je geen homografie vinden omdat er meerdere vlakken zijn.



- there is no 2D homography that relates these two images of a non-planar scene: the depth of each point is needed to map the left image onto the right image
- some 3D points visible at the left may be occluded at the right because of **parallax** (and vice versa)