B4 Computational Geometry

David Murray

david.murray@eng.ox.ac.uk
www.robots.ox.ac.uk/∼dwm/Courses/3CG

Michaelmas 2006

Overview

Computational geometry is concerned with

- the derivation of techniques
- the design of efficient algorithms and
- the construction of effective representations

for geometric computation.

Techniques from computational geometry are used in:

- Computer Graphics
- Computer Aided Design
- Computer Vision
- Robotics

Topics

- **Lecture 1**: Euclidean, similarity, affine and projective transformations. Homogeneous coordinates and matrices. Coordinate frames. Perspective projection and its matrix representation.
- **Lecture 2**: Perspective projection and its matrix representation. Vanishing points. Applications of projective transformations.
- **Lecture 3**: Convexity of point-sets, convex hull and algorithms. Conics and quadrics, implicit and parametric forms, computation of intersections.
- **Lecture 4**: Bezier curves, B-splines. Tensor-product surfaces.

Useful Texts

Bartels, Beatty and Barsky, “An introduction to splines for use in computer graphics and geometric modeling”, Morgan Kaufmann, 1987. Everything you could want to know about splines.

Faux and Pratt, “Computational geometry for design and manufacture”, Ellis Horwood, 1979. Good on curves and transformations.

Farin, “Curves and Surfaces for Computer-Aided Geometric Design : A Practical Guide”, Academic Press, 1996.

Foley, van Dam, Feiner and Hughes, “Computer graphics - principles and practice”, Addison Wesley, second edition, 1995. *The* computer graphics book. Covers curves and surfaces well.

Hartley and Zisserman “Multiple View Geometry in Computer Vision”, CUP, 2000. Chapter 1 is a good introduction to projective geometry.

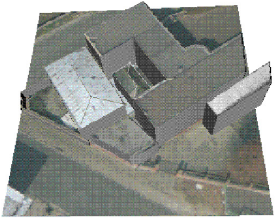
O’Rourke, “Computational geometry in C”, CUP, 1998. Very straightforward to read, many examples. Highly recommended.

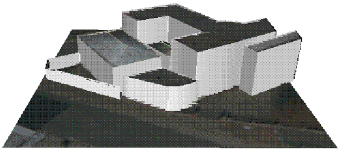
Preparata and Shamos, “Computational geometry, an introduction”, Springer-Verlag, 1985. Very formal and complete for particular algorithms.

Example I: Virtual Reality Models from Images

Input: Four overlapping aerial images of the same urban scene

Objective: Texture mapped 3D models of buildings



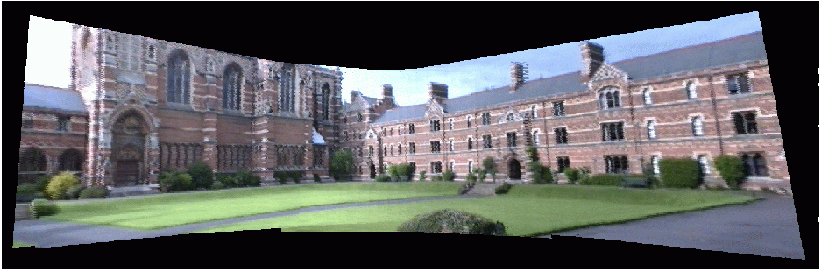


Example II: Video Mosaicing



Example II: Video Mosaicing

Example II: Video Mosaicing



Lecture 1.

Lecture 1:

Transformations, Homogeneous Coordinates, and Coordinate Frames

- Euclidean, similarity, affine and projective transformations.
- Homogeneous coordinates and matrices.
- Coordinate frames.

Hierarchy of transformations

We will look at **linear transformations** represented by matrices of **increasing** generality:

Euclidean $\rightarrow$
Similarity $\rightarrow$
Affine $\rightarrow$
Projective

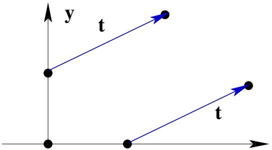
We consider both

- $2D \rightarrow 2D$ mappings ("plane to plane" mappings); and
- $3D \rightarrow 3D$ transformations

Class I: Euclidean transformations: translation & rotation

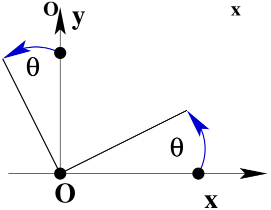
1. *Translation* — 2 dof in 2D

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$



2. *Rotation* — 1 dof in 2D

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$



In vector notation, a Euclidean transformation is written

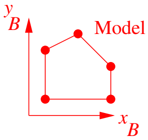
$$\mathbf{x}' = \boldsymbol{R}\mathbf{x} + \mathbf{t}$$

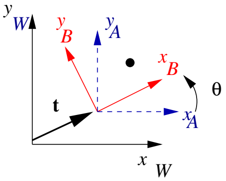
$\boldsymbol{R}$ is the **orthogonal** rotation matrix, $\boldsymbol{R}\boldsymbol{R}^\top = \boldsymbol{I}$, and $\mathbf{x}'$ etc are column vectors.

Build transformations in steps ...

- Often useful to introduce intermediate coordinate frames. For example:

Object model described in body-centered frame B
Pose $(\theta, \mathbf{t})$ of model frame given w.r.t. world frame W
Where is $\mathbf{x}_B$ in W?





- In an "aligned" frame $\mathbf{x}_A = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \mathbf{x}_B$.
  Check using the point $(1, 0)$*. It* should *be* $(+\cos\theta, +\sin\theta)$ *in the A frame.*
- Then $\mathbf{x}_W = \mathbf{x}_A + \mathbf{t}_{\text{Origin of B in W}}$.
  Check the above using the origin of A. It should *be* $\mathbf{t}_{OBW}$ *in W frame ...*

In 3D ...

In 3D the transformation $\mathbf{X}' = \boldsymbol{R}_{3\times3}\mathbf{X} + \mathbf{T}$ has 6 dof. Two major ways of representing 3 rotation

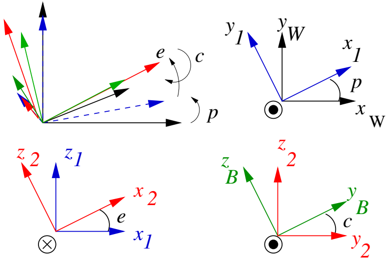
- rotation about successive new axes: eg ZYX Euler angles
- rotation about “old fixed axes”: eg ZXY roll-pitch-yaw

In each case the order is important, as rotations do not commute.

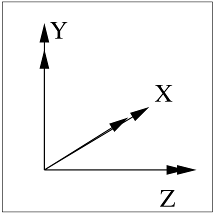
$$\mathbf{X}_W = \begin{bmatrix} \cos p & -\sin p & 0 \\ \sin p & \cos p & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{X}_1$$

$$\mathbf{X}_1 = \begin{bmatrix} \cos e & 0 & -\sin e \\ 0 & 1 & 0 \\ \sin e & 0 & \cos e \end{bmatrix} \mathbf{X}_2$$

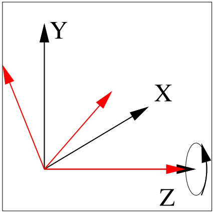
$$\mathbf{X}_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos c & -\sin c \\ 0 & \sin c & \cos c \end{bmatrix} \mathbf{X}_B$$



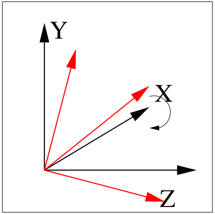
Rotation about “old fixed axes”: eg ZXY roll-pitch-yaw



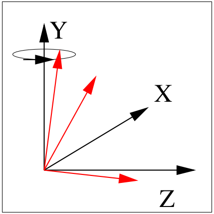
Start



Roll about ORIG Z



Pitch about orig X

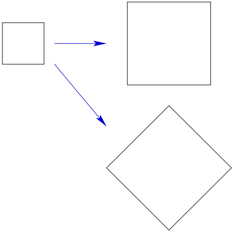


Yaw about orig Y

Class II: Similarity transformations

- A **Euclidean transformation** is an **isometry** — an action that preserves lengths and angles.
- Apply an isotropic scaling *s* to an isometry isotropic scaling and you'll arrive at a **similarity** transformation.
- A **similarity** had 4 degrees of freedom in 2D

$$\mathbf{x}' = s\boldsymbol{R}\mathbf{x} + \mathbf{t}$$



- A similarity preserves
  - **ratios of lengths**
  - **ratios of areas**, and
  - **angles**.
- It is the most general transformation that preserves "**shape**".

Class III: Affine transformations

- An **affine transformation** (6 degrees of freedom in 2D) — is a non-singular linear transformation followed by a translation:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} & \boldsymbol{A} & \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

with $\boldsymbol{A}$ a 2×2 non-singular matrix.

- In vector form:

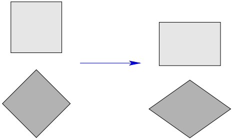
$$\mathbf{x}' = \boldsymbol{A}\mathbf{x} + \mathbf{t}$$

- Angles and length ratios are **not** preserved.
- How many points required to determine an affine transform in 2D?

Examples of affine transformations

1. Both the previous classes: Euclidean, similarity.
2. Scalings in the x and y directions

$$\boldsymbol{A} = \begin{bmatrix} \mu_1 & 0 \\ 0 & \mu_2 \end{bmatrix}$$



This is non-isotropic if $\mu_1 \equiv \mu_2$.

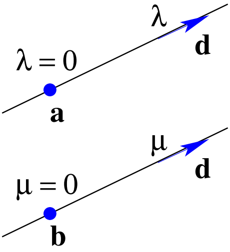
3. If $\boldsymbol{A}$ is a symmetric matrix.
   then $\boldsymbol{A}$ can be decomposed as: *(it's an eigen-decomposition)*

$$\boldsymbol{A} = \boldsymbol{R}\,\boldsymbol{D}\,\boldsymbol{R}^\top = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

where λ_1 and λ_2 are its eigenvalues. i.e. scalings in two dirns rotated by θ.

Affine transformations map parallel lines to ...

- Always useful to think what is preserved in a transformation ...



$$\begin{aligned}
\mathbf{x}_{\mathrm{A}}(\lambda) &= \mathbf{a} + \lambda \mathbf{d} \\
\mathbf{x}_{\mathrm{B}}(\lambda) &= \mathbf{b} + \mu \mathbf{d} \\
\mathbf{x}' &= \boldsymbol{A}\mathbf{x} + \mathbf{t}
\end{aligned}$$

$$\begin{aligned}
\Rightarrow \mathbf{x}'_{\mathrm{A}}(\lambda) &= \boldsymbol{A}(\mathbf{a} + \lambda \mathbf{d}) + \mathbf{t} \\
&= (\boldsymbol{A}\mathbf{a} + \mathbf{t}) + \lambda(\boldsymbol{A}\mathbf{d}) \\
&= \mathbf{a}' + \lambda \mathbf{d}' \\
\mathbf{x}'_{\mathrm{B}}(\mu) &= \boldsymbol{A}(\mathbf{b} + \mu \mathbf{d}) + \mathbf{t} \\
&= (\boldsymbol{A}\mathbf{b} + \mathbf{t}) + \mu(\boldsymbol{A}\mathbf{d}) \\
&= \mathbf{b}' + \mu \mathbf{d}'
\end{aligned}$$

- Lines are still parallel – they both have direction $\mathbf{d}'$.
- Affine transformations also preserve ...

Homogeneous notation — motivation

- If the translation $\mathbf{t}$ is zero, then transformations can be **concatenated** by simple matrix multiplication:

$$\mathbf{x}_1 = \boldsymbol{A}_1\mathbf{x} \quad \text{and} \quad \mathbf{x}_2 = \boldsymbol{A}_2\mathbf{x}_1 \quad \text{THEN} \quad \mathbf{x}_2 = \boldsymbol{A}_2\boldsymbol{A}_1\mathbf{x}$$

- However, if the translation is non-zero it becomes a mess

$$\begin{aligned}
\mathbf{x}_1 &= \boldsymbol{A}_1\mathbf{x} + \mathbf{t}_1 \\
\mathbf{x}_2 &= \boldsymbol{A}_2\mathbf{x}_1 + \mathbf{t}_2 \\
&= \boldsymbol{A}_2(\boldsymbol{A}_1\mathbf{x} + \mathbf{t}_1) + \mathbf{t}_2 \\
&= (\boldsymbol{A}_2\boldsymbol{A}_1)\mathbf{x} + (\boldsymbol{A}_2\mathbf{t}_1 + \mathbf{t}_2)
\end{aligned}$$

- If instead 2D points $\begin{pmatrix} x \\ y \end{pmatrix}$ are represented by a three vector $\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$ then the transformation can be represented by a 3×3 matrix ...

Homogeneous notation

The matrix has block form:

$$\begin{pmatrix} \mathbf{x}' \\ 1 \end{pmatrix} = \begin{bmatrix} \boldsymbol{A} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix} = \left[\begin{array}{cc:c} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ \hdashline 0 & 0 & 1 \end{array}\right] \begin{pmatrix} x \\ y \\ \hdashline 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{A}\mathbf{x} + \mathbf{t} \\ 1 \end{pmatrix}$$

Transformations can now **ALWAYS** be concatenated by matrix multiplication

$$\begin{aligned}
\begin{pmatrix} \mathbf{x}_1 \\ 1 \end{pmatrix} &= \begin{bmatrix} \boldsymbol{A}_1 & \mathbf{t}_1 \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{A}_1\mathbf{x} + \mathbf{t}_1 \\ 1 \end{pmatrix} \\
\begin{pmatrix} \mathbf{x}_2 \\ 1 \end{pmatrix} &= \begin{bmatrix} \boldsymbol{A}_2 & \mathbf{t}_2 \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{pmatrix} \mathbf{x}_1 \\ 1 \end{pmatrix} = \begin{bmatrix} \boldsymbol{A}_2 & \mathbf{t}_2 \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{A}_1 & \mathbf{t}_1 \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix} \\
&= \begin{bmatrix} \boldsymbol{A}_2\boldsymbol{A}_1 & \boldsymbol{A}_2\mathbf{t}_1 + \mathbf{t}_2 \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix} = \begin{pmatrix} (\boldsymbol{A}_2\boldsymbol{A}_1)\mathbf{x} + (\boldsymbol{A}_2\mathbf{t}_1 + \mathbf{t}_2) \\ 1 \end{pmatrix}
\end{aligned}$$

Homogeneous notation — definition

- $\mathbf{x} = (x, y)^\top$ is represented in homogeneous coordinates by any **3-vector**

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

  such that

$$x = x_1/x_3 \qquad y = x_2/x_3$$

- So the following homogeneous vectors represent the same point for any $\lambda \equiv 0$.

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \text{ and } \begin{pmatrix} \lambda x_1 \\ \lambda x_2 \\ \lambda x_3 \end{pmatrix}$$

- For example, the homogeneous vectors $(2, 3, 1)^\top$ and $(4, 6, 2)^\top$ represent the **same** inhomogeneous point $(2, 3)^\top$

Homogeneous notation – rules for use

- Then the rules for using homogeneous coordinates for transformations are

1. Convert the inhomogeneous point to an homogeneous vector:

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

2. Apply the 3×3 matrix transformation.
3. Dehomogenise the resulting vector:

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \rightarrow \begin{pmatrix} x_1/x_3 \\ x_2/x_3 \end{pmatrix}$$

NB the matrix needs only to be defined up to scale.
E.g.

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 0 & 0 & 1 \end{bmatrix} \text{ and } \begin{bmatrix} 2 & 4 & 6 \\ 8 & 10 & 12 \\ 0 & 0 & 2 \end{bmatrix}$$

represent the SAME 2D affine transformation
Think about degrees of freedom ...

Homogeneous notation for $\mathcal{R}^3$

- A point

$$\mathbf{X} = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

is represented by a homogeneous 4-vector:

$$\begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{pmatrix}$$

such that

$$X = \frac{X_1}{X_4} \qquad Y = \frac{X_2}{X_4} \qquad Z = \frac{X_3}{X_4}$$

Example: The Euclidean transformation in 3D

$$\mathbf{X}' = \boldsymbol{R}\mathbf{X} + \mathbf{T}$$

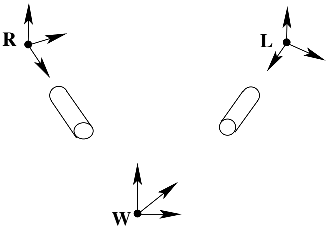
where $\boldsymbol{R}$ is a 3×3 rotation matrix, and $\mathbf{T}$ a translation 3-vector, is represented as

$$\begin{pmatrix} X_1' \\ X_2' \\ X_3' \\ X_4' \end{pmatrix} = \begin{bmatrix} \boldsymbol{R} & \mathbf{T} \\ \mathbf{0}^\top & 1 \end{bmatrix}_{4\times 4} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} \boldsymbol{R} & \mathbf{T} \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}$$

with

$$\mathbf{X}' = \begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} = \frac{1}{X_4'} \begin{pmatrix} X_1' \\ X_2' \\ X_3' \end{pmatrix}$$

Application to coordinate frames: Eg - stereo cameras

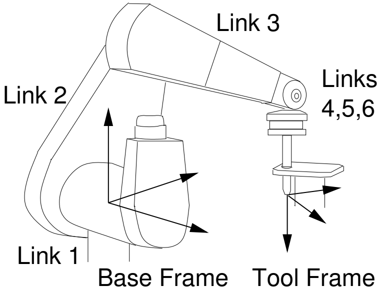


$$\begin{pmatrix} \mathbf{X}_{\mathrm{R}} \\ 1 \end{pmatrix} = \begin{bmatrix} \boldsymbol{R}_{\mathrm{RW}} & \mathbf{T}_{\mathrm{RW}} \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \begin{pmatrix} \mathbf{X}_{\mathrm{W}} \\ 1 \end{pmatrix} \qquad \begin{pmatrix} \mathbf{X}_{\mathrm{L}} \\ 1 \end{pmatrix} = \begin{bmatrix} \boldsymbol{R}_{\mathrm{LW}} & \mathbf{T}_{\mathrm{LW}} \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \begin{pmatrix} \mathbf{X}_{\mathrm{W}} \\ 1 \end{pmatrix}$$

Then

$$\begin{pmatrix} \mathbf{X}_{\mathrm{R}} \\ 1 \end{pmatrix} = \begin{bmatrix} \boldsymbol{R}_{\mathrm{RW}} & \mathbf{T}_{\mathrm{RW}} \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{R}_{\mathrm{LW}} & \mathbf{T}_{\mathrm{LW}} \\ \mathbf{0}^{\top} & 1 \end{bmatrix}^{-1} \begin{pmatrix} \mathbf{X}_{\mathrm{L}} \\ 1 \end{pmatrix} = \begin{bmatrix} 4 \times 4 \end{bmatrix} \begin{pmatrix} \mathbf{X}_{\mathrm{L}} \\ 1 \end{pmatrix}$$

Application to coordinate frames: Eg - Puma robot arm



Kinematic chain:

$$\begin{pmatrix} \mathbf{X}_{\mathrm{T}} \\ 1 \end{pmatrix} = \begin{bmatrix} \boldsymbol{R}_{\mathrm{T6}} & \mathbf{T}_{\mathrm{T6}} \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \cdots \begin{bmatrix} \boldsymbol{R}_{32} & \mathbf{0} \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{R}_{21} & \mathbf{T}_{21} \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{R}_{1\mathrm{B}} & \mathbf{T}_{1\mathrm{B}} \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}$$

$$= \begin{bmatrix} 4 \times 4 \end{bmatrix} \begin{pmatrix} \mathbf{X}_{\mathrm{B}} \\ 1 \end{pmatrix}$$

A note on the inverse ...

- It must be the case that

$$\begin{bmatrix} \mathbf{R}_{AB} & \mathbf{T}_{AB} \\ \mathbf{0}^\top & 1 \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{R}_{BA} & \mathbf{T}_{BA} \\ \mathbf{0}^\top & 1 \end{bmatrix}$$

- Now, we know that

$$\mathbf{R}_{BA} = \mathbf{R}_{AB}^{-1}$$

  but what is $\mathbf{T}_{BA}$?

- Tempting to say $-\mathbf{T}_{AB}$, but no.

$$\begin{aligned} \mathbf{X}_A &= \mathbf{R}_{AB}\mathbf{X}_B + \mathbf{T}_{AB} && (\mathbf{T}_{AB} \text{ is Origin of B in A}) \\ \Rightarrow \mathbf{X}_B &= \mathbf{R}_{BA}(\mathbf{X}_A - \mathbf{T}_{AB}) \\ \Rightarrow \mathbf{X}_B &= \mathbf{R}_{BA}\mathbf{X}_A - \mathbf{R}_{BA}\mathbf{T}_{AB} \\ \text{BUT } \mathbf{X}_B &= \mathbf{R}_{BA}\mathbf{X}_A + \mathbf{T}_{BA} && (\mathbf{T}_{BA} \text{ is Origin of A in B}) \\ \Rightarrow \mathbf{T}_{BA} &= -\mathbf{R}_{BA}\mathbf{T}_{AB} \end{aligned}$$

Class IV: Projective transformations

- A projective transformation is a linear transformation on homogeneous n-vectors represented by a non-singular $n \times n$ matrix.
- **In 2D**

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

- Note the difference from an affine transformation is only in the first two elements of the last row.
- In inhomogeneous (normal) notation, a projective transformation is a **non-linear** map

$$x' = \frac{x'_1}{x'_3} = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}, \qquad y' = \frac{x'_2}{x'_3} = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}$$

- The 3×3 matrix has 8 dof ...

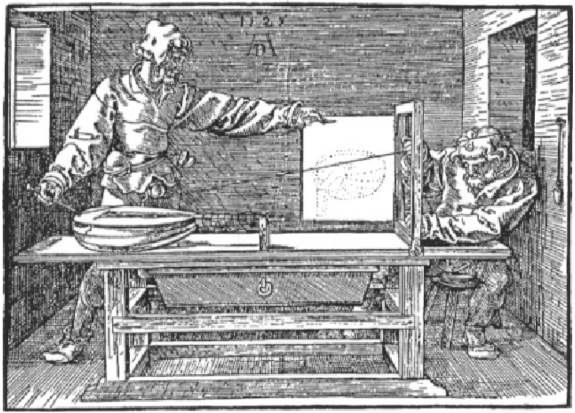
Class IV: 3D-3D Projective transformations

- **In 3D**

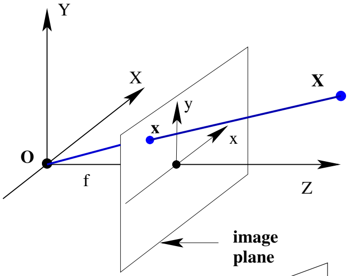
$$\begin{pmatrix} X'_1 \\ X'_2 \\ X'_3 \\ X'_4 \end{pmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \\ p_{41} & p_{42} & p_{43} & p_{44} \end{bmatrix} \begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{pmatrix}$$

- The 4×4 matrix has 15 dof ...

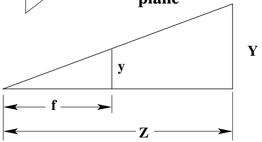
Perspective is a subclass of projective transformation



Perspective (central) projection — 3D to 2D



- Mathematical idealized camera 3D $\rightarrow$ 2D
- Image coordinates xy
- Camera frame XYZ (origin at optical centre)
- Focal length f, image plane is at $Z = f$.
- Use similar triangles

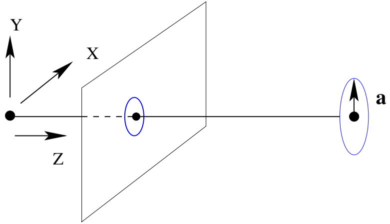


$$\frac{x}{f} = \frac{X}{Z} \qquad \frac{y}{f} = \frac{Y}{Z} \qquad \text{or} \quad \mathbf{x} = f\frac{\mathbf{X}}{Z}$$

where $\mathbf{x}$ and $\mathbf{X}$ are **3-vectors**, with $\mathbf{x} = (x, y, f)^\top$, $\mathbf{X} = (X, Y, Z)^\top$.

Examples

Circle in space, orthogonal to and centred on the Z-axis:



$$\begin{aligned}\mathbf{X}(\theta) &= (a\cos\theta, a\sin\theta, Z)^\top \\ \mathbf{x}(\theta) &= (\frac{fa}{Z}\cos\theta, \frac{fa}{Z}\sin\theta, f)^\top \\ \Rightarrow (x, y) &= \frac{fa}{Z}(\cos\theta, \sin\theta)\end{aligned}$$

Image is a circle of radius fa/Z
— inverse distance scaling

Now move circle in X direction:

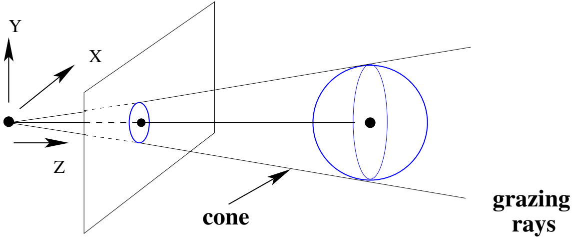
that is, to $\mathbf{X}_1(\theta) = (a\cos\theta + X_0, a\sin\theta, Z)^\top$

Exercise:

What happens to the image? Is it still a circle? Is it larger or smaller?

Examples ctd/

Sphere concentric with Z-axis:



Intersection of **cone** with image plane is a circle.

Exercise:

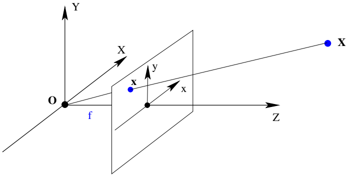
Now move sphere in the X direction. What happens to the image?

The Homogeneous 3 × 4 Projection Matrix

- In inhomogeneous coords

$$\mathbf{x} = f\mathbf{X}/Z$$

Choose $f = 1$ from now on.



- Homogeneous image coordinates $(x_1, x_2, x_3)^\top$ represent $\mathbf{x} = \mathbf{X}/Z$ if

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = [\boldsymbol{I} \mid \mathbf{0}] \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}$$

- Check that $x = x_1/x_3 = X/Z \qquad y = x_2/x_3 = Y/Z$
- Then perspective projection is a linear map, represented by a 3 × 4 **projection matrix**, from 3D to 2D.

Example: a 3D point

- Non-homogeneous $\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 6 \\ 4 \\ 2 \end{pmatrix}$ is imaged at $(x, y) = (6/2, 4/2) = (3, 2)$.
- In homogeneous notation using 3×4 projection matrix:

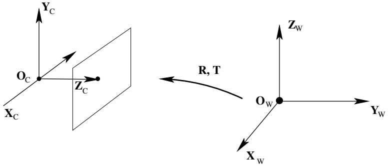
$$\begin{pmatrix} \lambda x \\ \lambda y \\ \lambda \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} 6 \\ 4 \\ 2 \\ 1 \end{pmatrix} = \begin{pmatrix} 6 \\ 4 \\ 2 \end{pmatrix}$$

which is the 2D inhomogeneous point $(x, y) = (3, 2)$.

Suppose scene is describe in a World coord frame

- The Euclidean transformation between the camera and world coordinate frames is $\mathbf{X}_{\mathrm{C}} = \boldsymbol{R}\mathbf{X}_{\mathrm{W}} + \mathbf{T}$:

$$\begin{pmatrix} \mathbf{X}_{\mathrm{C}} \\ 1 \end{pmatrix} = \begin{bmatrix} \boldsymbol{R} & \mathbf{T} \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \begin{pmatrix} \mathbf{X}_{\mathrm{W}} \\ 1 \end{pmatrix}$$



- Concatenating the two matrices ...

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{R} & \mathbf{T} \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \begin{pmatrix} \mathbf{X}_{\mathrm{W}} \\ 1 \end{pmatrix} = [\, \boldsymbol{R} \mid \mathbf{T}] \begin{pmatrix} \mathbf{X}_{\mathrm{W}} \\ 1 \end{pmatrix}$$

which defines the 3×4 projection matrix $\boldsymbol{P} = [\boldsymbol{R}| \ \mathbf{T}]$ from a Euclidean World coordinate frame to an image.

Suppose scene described as set of Objects and Poses

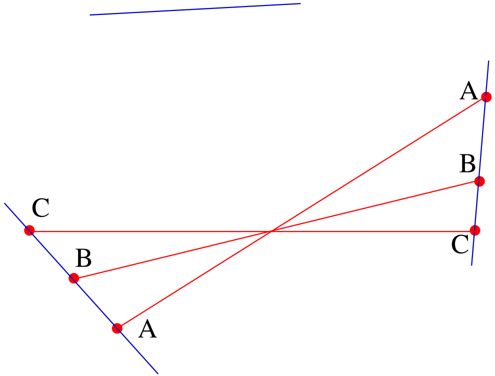
- Now each 3D object O is described in it own Object frame ...
- Each Object frame is given a Pose $[\boldsymbol{R}_o, \mathbf{T}_o]$ relative to World frame ...
- Cameras are placed at $[\boldsymbol{R}_c, \mathbf{T}_c]$ relative to world frame ...

$$\begin{pmatrix} \mathbf{x}_c \\ 1 \end{pmatrix} = \boldsymbol{K}_c \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{R}_c & \mathbf{T}_c \\ \mathbf{0}^\top & 1 \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{R}_o & \mathbf{T}_o \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{pmatrix} \mathbf{X}_o \\ 1 \end{pmatrix}$$

- 3×3 matrix $\boldsymbol{K}_c$ allows each camera to have a different focal length etc ...
- You can now do 3D computer graphics ...

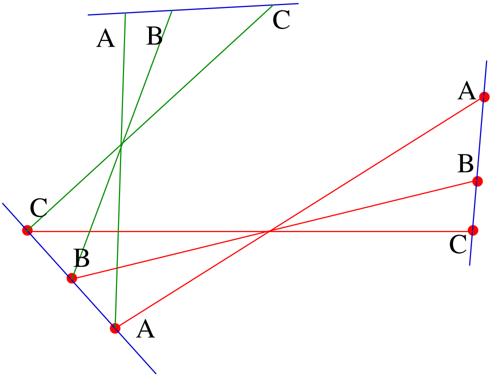
Isn't every projective transform a perspective projection?

- A projective trans followed by a projective trans is a ....
- So a perspective trans followed by a perpspective trans is a ....



Isn't every projective transform a perspective projection?

- A projective trans followed by a projective trans is a ....
- So a perspective trans followed by a perpspective trans is a ....



Isn't every projective transform a perspective projection?

- A projective trans followed by a projective trans is a ....
- So a perspective trans followed by a perpspective trans is a ....