Computational Geometry

4 Lectures — Michaelmas Term 2003

1 Tutorial Sheet — Dr ID Reid

Overview

Computational geometry is concerned with efficient algorithms and representations for geometric computation.

Techniques from computational geometry are used in:

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

Topics 0.1

- 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.

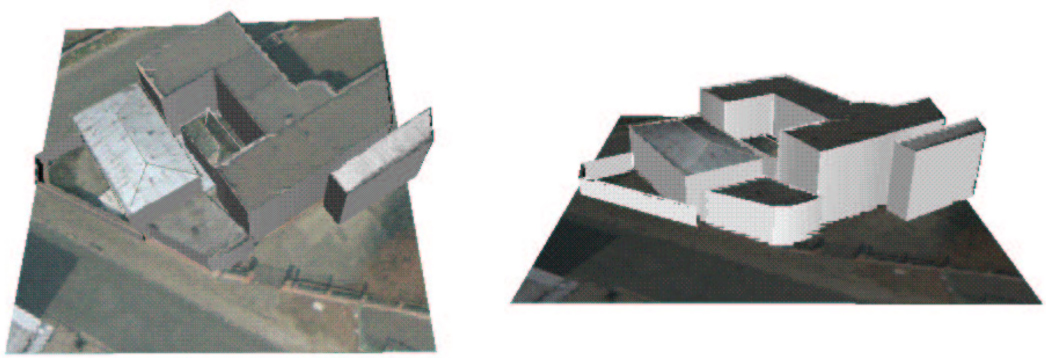
Useful Texts 0.2

- **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 0.3

Input: Four overlapping aerial images of the same urban scene

Objective: Texture mapped 3D models of buildings



1: Transformations, Homogeneous Coordinates, and Coordinate Frames

Topics 1.1

- **Lecture 1: Euclidean, similarity, affine and projective transformations. Homogeneous coordinates and matrices. Coordinate frames.**
- 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.

Hierarchy of transformations 1.2

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

- Euclidean → Similarity → Affine → projective.

Consider both

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

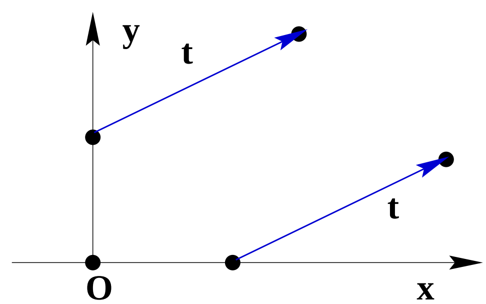
as well as

- $3D \to 2D$ mappings ("projections")

Class I: Euclidean transformations: translation & rotation 1.3

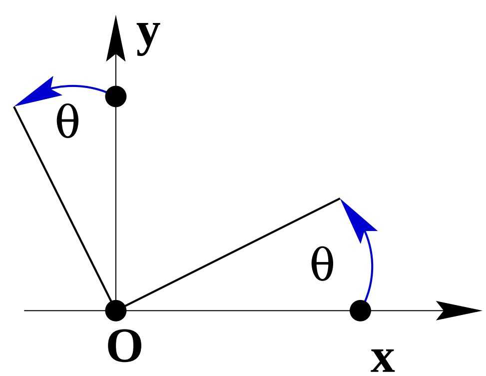
1. *Translation* — 2 dof in 2D

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



2. *Rotation* — 1 dof in 2D

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



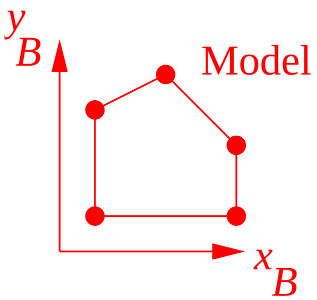
In vector notation, a Euclidean transformation is written

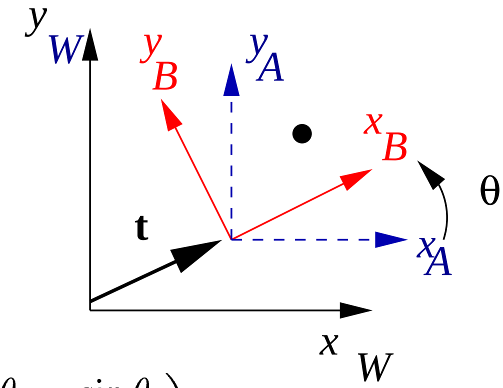
$$\mathbf{x}' = \mathtt{R}\mathbf{x} + \mathbf{t}$$

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

Build transformations in steps ...

Often useful to introduce intermediate coordinate frames.
Example: Object model described in body-centered coord frame. Pose $(\theta, \mathbf{t})$ of model frame given w.r.t. world coord frame. Where is $\mathbf{x}_B$ in the world?





$$\mathbf{x}_A = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \mathbf{x}_B$$

Check the above using the point $(1, 0)$. It *should* be $(+\cos\theta, +\sin\theta)$ in the A frame.

$$\mathbf{x}_W = \mathbf{x}_A + \mathbf{t}_{\text{Origin of B in W}}$$

Check the above using the origin of A. It *should* be $\mathbf{t}_{OBW}$ in W frame ...

In 3D ... 1.5

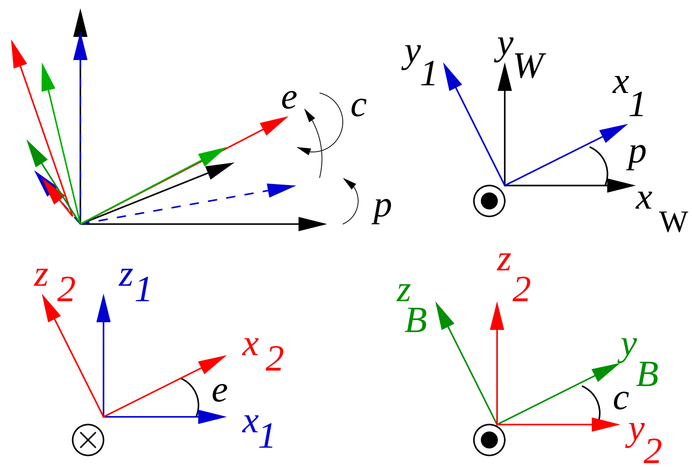
In 3D the transformation $\mathbf{X}' = \mathtt{R}_{3\times 3}\mathbf{X} + \mathbf{T}$ has 6 dof.

Two major ways of defining 3D rotation:
(i) rotation about successive new axes: eg YXY pan-tilt-verge, or XYZ tilt-pan-cyclorotation
(ii) rotation about "old fixed axes": eg ZXY roll-pitch-yaw
In each case the order is important, as rotations do not commute.

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

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

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



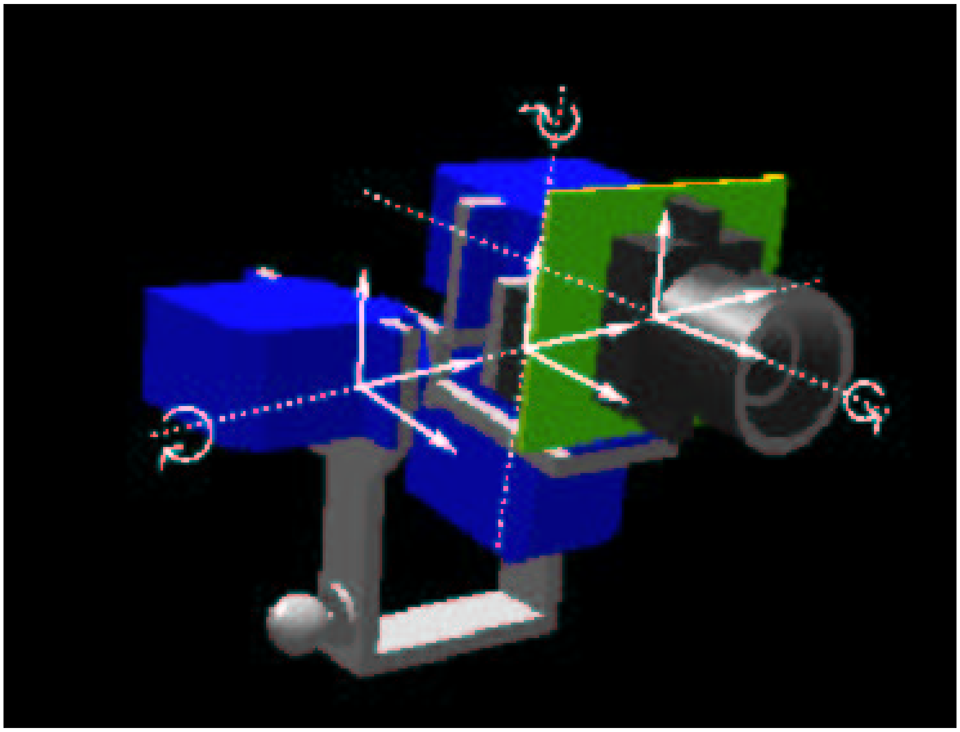
Rotation about an axis 1.6

The rotation matrix corresponding to a rotation of an angle θ about an axis with unit length $\mathbf{a} = [a_x, a_y, a_z]^T$ is given by:

$$R = \begin{bmatrix} \cos\theta + (1-\cos\theta)a_x^2 & (1-\cos\theta)a_xa_y - \sin\theta a_z & (1-\cos\theta)a_xa_z + \sin\theta a_y \\ (1-\cos\theta)a_xa_y + \sin\theta a_z & \cos\theta + (1-\cos\theta)a_y^2 & (1-\cos\theta)a_ya_z - \sin\theta a_x \\ (1-\cos\theta)a_xa_z - \sin\theta a_y & (1-\cos\theta)a_ya_z + \sin\theta a_x & \cos\theta + (1-\cos\theta)a_z^2 \end{bmatrix}$$

Conversely, for a given rotation matrix R, the direction of the axis is given by the eqigenvector corresponding to the unit eigenvalue, and the angle by the solution to trace$(R) = 2\cos\theta + 1$.

Consider a pan-tilt device:



1. Tilt: rotate by angle t about x-axis

$$R_{B1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos t & -\sin t \\ 0 & \sin t & \cos t \end{bmatrix}$$

2. Pan: rotate by angle p about *new* y-axis

$$y' = R_x(t)y = [0, \cos t, \sin t]^T$$

$$R_{21} = R_{y'} = \begin{bmatrix} \cos p & -\sin p \sin t & \sin p \cos t \\ \sin p \sin t & \cos p + (1 - \cos p)\cos^2 t & (1 - \cos p)\cos t \sin t \\ -\sin p \cos t & (1 - \cos p)\cos t \sin t & \cos p + (1 - \cos p)\sin t^2 \end{bmatrix}$$

Hence

$$R_{2B} = R_{21}R_{1B} = \begin{bmatrix} \cos p & 0 & \sin p \\ \sin t \sin p & \cos t & -\sin t \cos p \\ -\cos t \sin p & \sin t & \cos t \cos p \end{bmatrix}$$

A little thought suggests an alternative derivation of R_{2B}.

Start at the *end* of the kinematic chain when all axes in their rest positions.

Now

1. Rotate *pan* axis around fixed y-axis

$$\begin{bmatrix} \cos p & 0 & \sin p \\ 0 & 1 & 0 \\ -\sin p & 0 & \cos p \end{bmatrix}$$

2. Rotate *tilt* axis around x-axis *which was unaffected by previous rotation*

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos t & -\sin t \\ 0 & \sin t & \cos t \end{bmatrix}$$

Yielding

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos t & -\sin t \\ 0 & \sin t & \cos t \end{bmatrix} \begin{bmatrix} \cos p & 0 & \sin p \\ 0 & 1 & 0 \\ -\sin p & 0 & \cos p \end{bmatrix} = \begin{bmatrix} \cos p & 0 & \sin p \\ \sin t \sin p & \cos t & -\sin t \cos p \\ -\cos t \sin p & \sin t & \cos t \cos p \end{bmatrix}$$

as before.

Class II: Similarity transformations 1.10

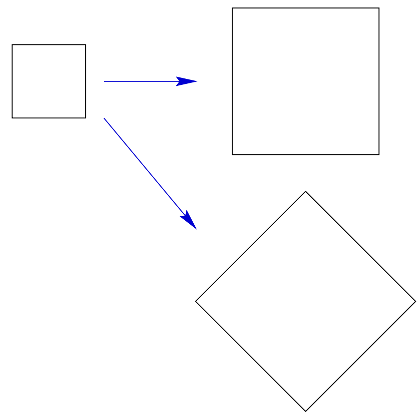
A Euclidean transformation is an isometry — an action that preserves lengths and angles.

An Isometry composed with isotropic scaling, s is called a similarity transformation.

A *similarity* — 4 degrees of freedom in 2D

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

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



Class III: Affine transformations 1.11

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

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

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

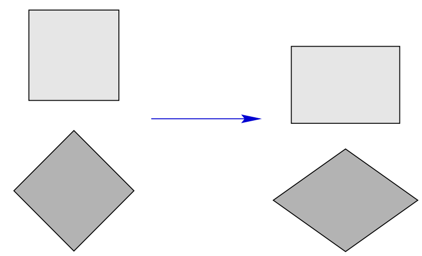
In vector form:

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

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

Examples of affine transformation 1.12

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



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

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

3. $\mathtt{A}$ a symmetric matrix.
Then $\mathtt{A}$ can be decomposed as: *it's an eigen-decomposition*

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

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

Affine transformations map parallel lines to parallel lines 1.13

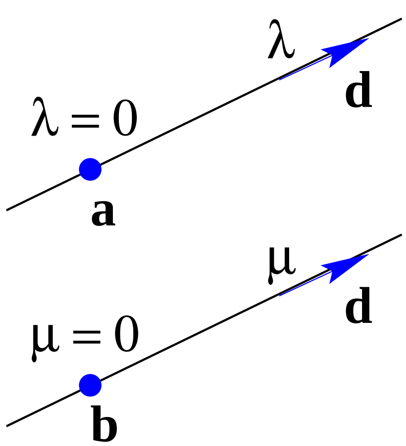
It is always useful to think what is preserved in a transformation ...

$$\begin{aligned} \mathbf{x}_\mathrm{A}(\lambda) &= \mathbf{a} + \lambda\mathbf{d} \\ \mathbf{x}_\mathrm{B}(\lambda) &= \mathbf{b} + \mu\mathbf{d} \end{aligned}$$

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

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

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



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:

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

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

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

Homogeneous notation

If 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 with **block form**:

$$\begin{pmatrix} \mathbf{x}' \\ 1 \end{pmatrix} = \begin{bmatrix} \mathtt{A} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & \vdots & t_x \\ a_{21} & a_{22} & \vdots & t_y \\ \dots & \dots & \vdots & \dots \\ 0 & 0 & \vdots & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ \dots \\ 1 \end{pmatrix} = \begin{pmatrix} \mathtt{A}\mathbf{x} + \mathbf{t} \\ 1 \end{pmatrix}$$

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

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

Homogeneous notation — definition 1.16

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

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

such that

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

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

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

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 1.17

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.
For example

$$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ and } \begin{bmatrix} 2 & 0 & 2 \\ 0 & 4 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

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

Homogeneous notation for $\mathcal{R}^3$ 1.18

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 1.19

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

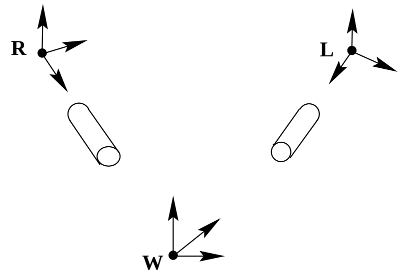
where $\mathtt{R}$ is a 3×3 rotation matrix, and $\mathbf{T}$ a translation 3-vector, is represented as

$$\begin{pmatrix} X'_1 \\ X'_2 \\ X'_3 \\ X'_4 \end{pmatrix} = \begin{bmatrix} \mathtt{R} & \mathbf{T} \\ \mathbf{0}^\top & 1 \end{bmatrix}_{4\times 4} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} \mathtt{R} & \mathbf{T} \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}$$

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: Example - stereo camera rig 1.20

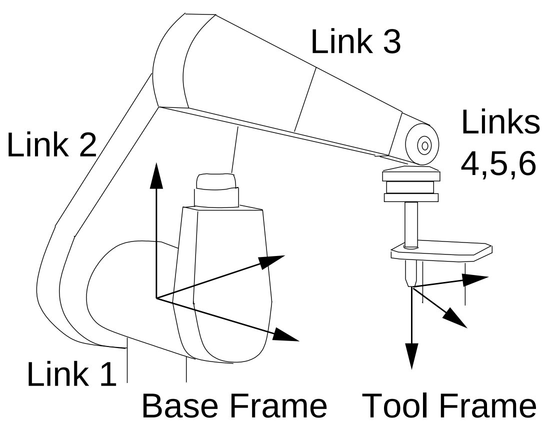


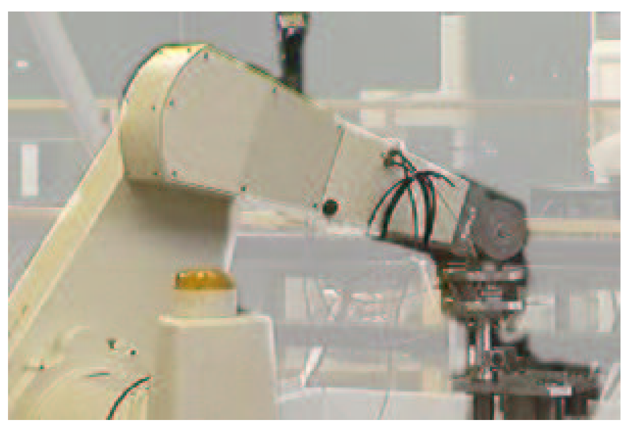
$$\begin{pmatrix} \mathbf{X}_{\mathrm{R}} \\ 1 \end{pmatrix} = \begin{bmatrix} \mathrm{R}_{\mathrm{RW}} & \mathbf{T}_{\mathrm{RW}} \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \begin{pmatrix} \mathbf{X}_{\mathrm{W}} \\ 1 \end{pmatrix} \qquad \begin{pmatrix} \mathbf{X}_{\mathrm{L}} \\ 1 \end{pmatrix} = \begin{bmatrix}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

Then

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

Application to coordinate frames: Example - Puma robot arm 1.21





Kinematic chain:

$$\begin{aligned} \begin{pmatrix} \mathbf{X}_{\mathrm{T}} \\ 1 \end{pmatrix} &= \begin{bmatrix} \mathrm{R}_{\mathrm{T6}} & \mathbf{T}_{\mathrm{T6}} \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \cdots \begin{bmatrix} \mathrm{R}_{32} & \mathbf{0} \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \begin{bmatrix} \mathrm{R}_{21} & \mathbf{T}_{21} \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \begin{bmatrix} \mathrm{R}_{1\mathrm{B}} & \mathbf{T}_{1\mathrm{B}} \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \begin{pmatrix} \mathbf{X}_{\mathrm{B}} \\ 1 \end{pmatrix} \\ &= \begin{bmatrix} & 4 \times 4 & \\ & & \end{bmatrix} \begin{pmatrix} \mathbf{X}_{\mathrm{B}} \\ 1 \end{pmatrix} \end{aligned}$$

A note on the inverse ... 1.22

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

Now,

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

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

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

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

Class IV: Projective transformations 1.23

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

2D — plane to plane

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

• 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 ...

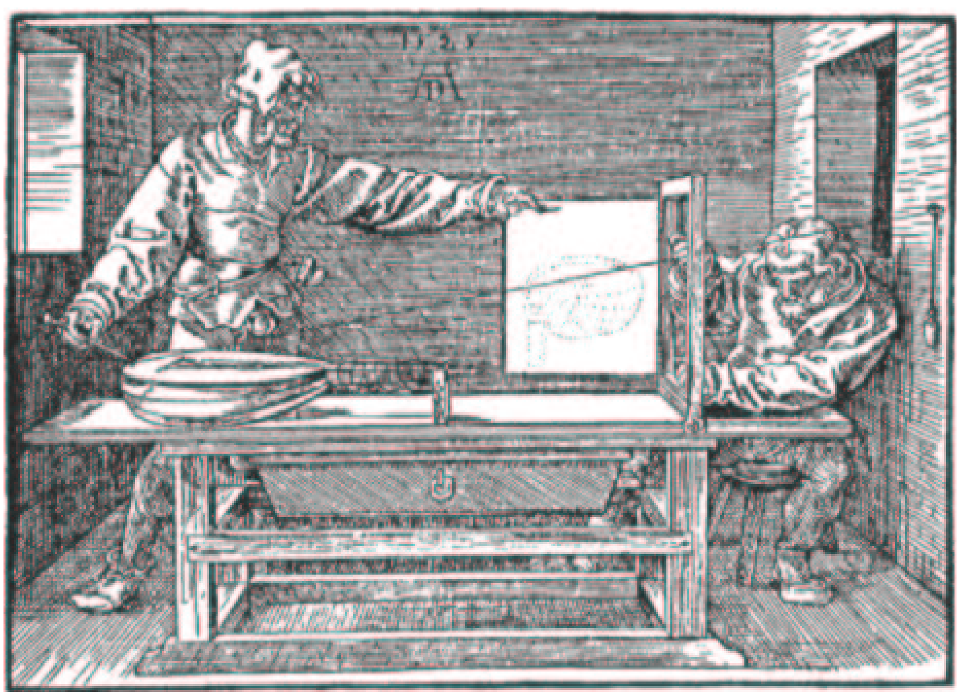
Class IV: 3D-3D Projective transformations

3D

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

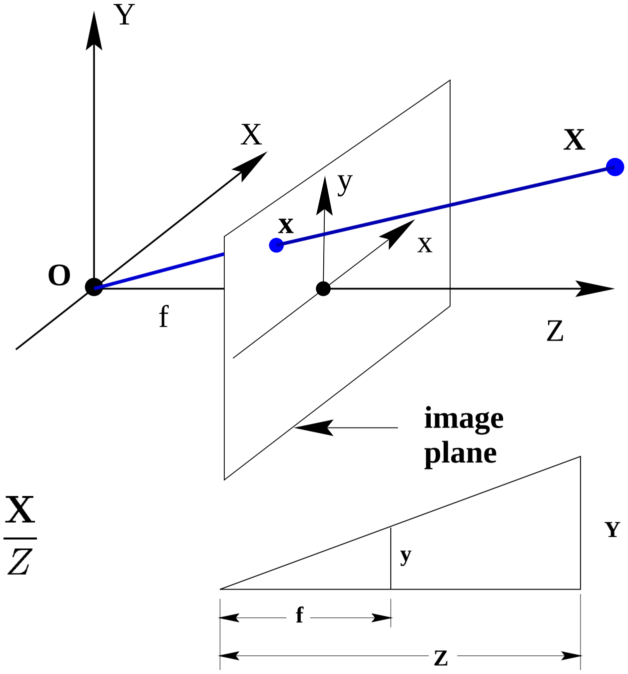
- The 4×4 matrix has 15 dof ...

Perspective projection is a subclass of projective transformation



Perspective (central) projection — 3D to 2D 1.26

The camera model Mathematical idealized camera 3D → 2D
- Image coordinates xy
- Camera frame XYZ (origin at optical centre)
- Focal length f, image plane is at $Z = f$.



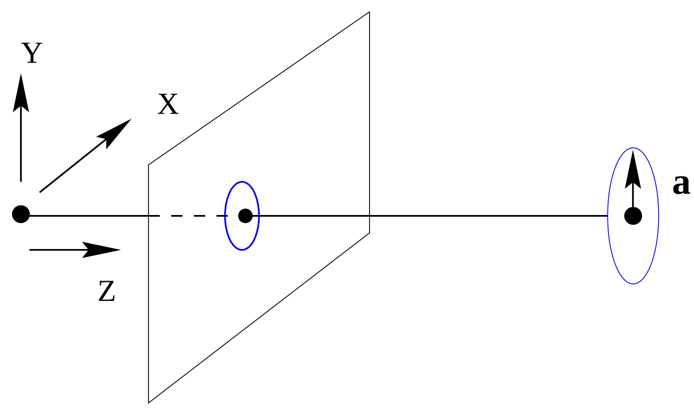
Similar triangles

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

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

Examples 1.27

1. 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

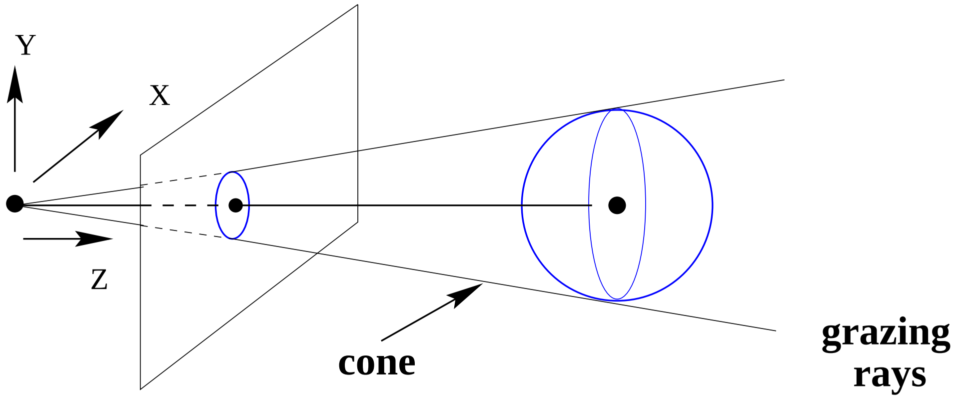
2. Now move circle in X direction:

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

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

Examples ctd/

3. Sphere concentric with Z-axis:



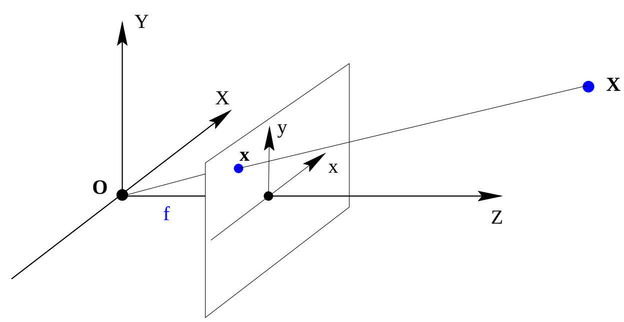
Intersection of **cone** with image plane is a circle.

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

The Homogeneous 3×4 Projection Matrix

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

Choose $f = 1$ from now on.



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

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

because then

$$x = \frac{x_1}{x_3} = \frac{X}{Z} \qquad y = \frac{x_2}{x_3} = \frac{Y}{Z}$$

Then perspective projection is a linear map, represented by a 3×4 **projection matrix**, from 3D to 2D.

Example: a 3D point

1.30

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:

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

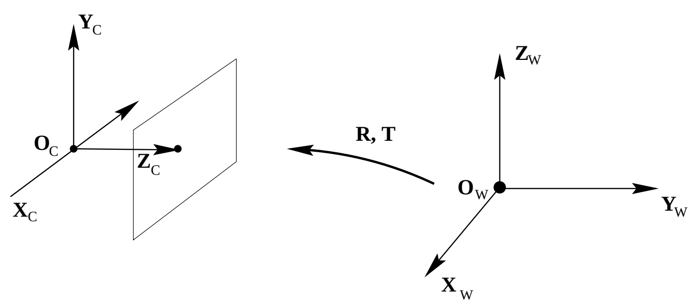
which is the 2D inhomogeneous point $(x, y) = (3, 2)$.

Supppose scene is describe in a World coord frame

1.31

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

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



Concatenating the two matrices ...

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

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

Suppose scene is described as set of Objects and Poses ... 1.32

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

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

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

Isn't every projective transformation a perspective projection? 1.33

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