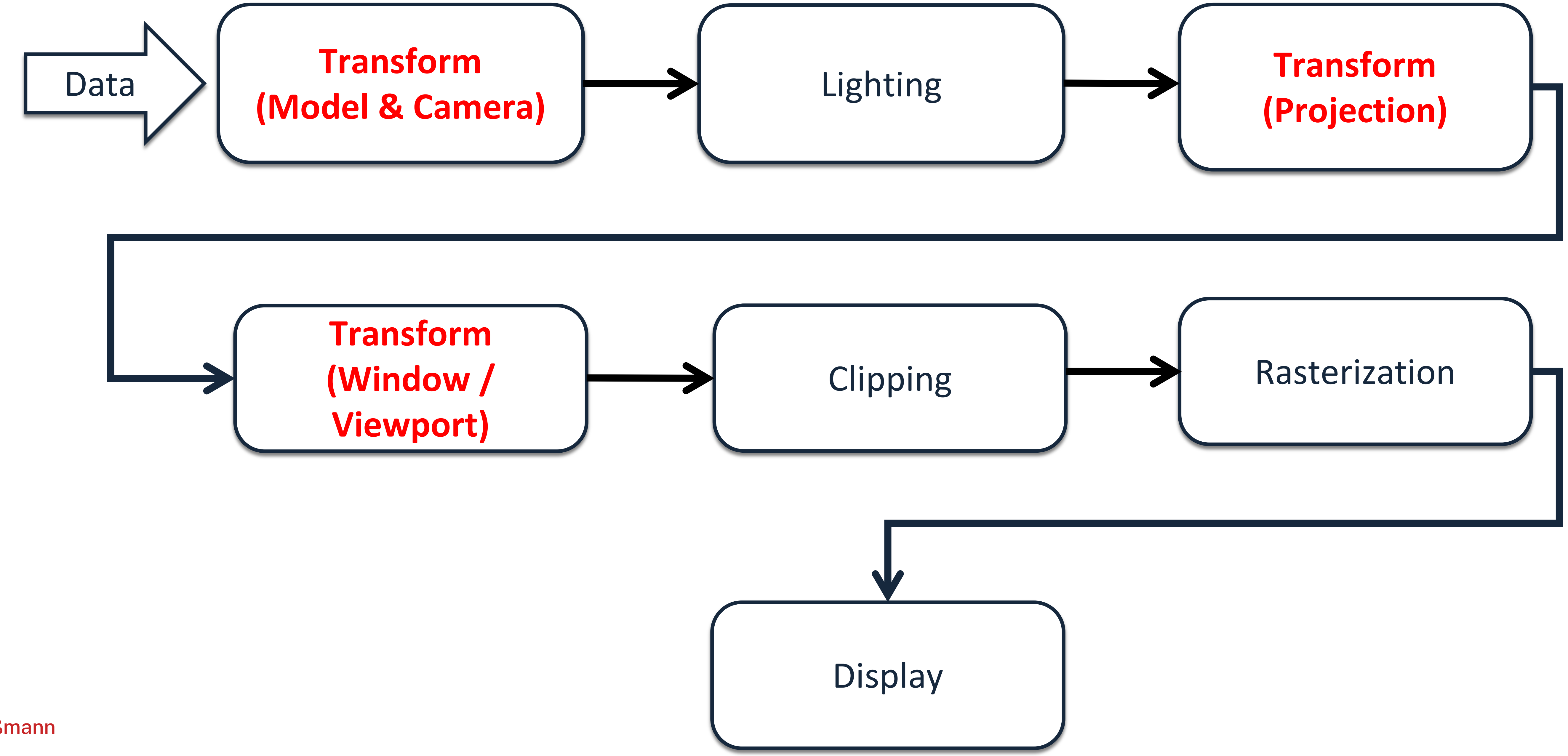
2.2 Transformations

Transformations

- Transformations are very important in CG and visualization!
  - Position objects in a scene (modeling)
  - Change shape of objects
  - Create copies of objects
  - Projections for virtual cameras
  - Position of projection on actual 2D window
  - Change of coordinate systems
  - Animations

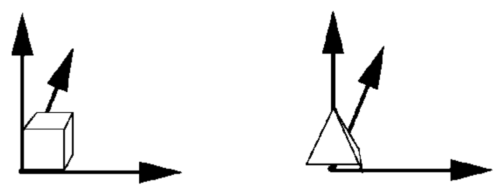
Transformations

Context: Rendering pipeline

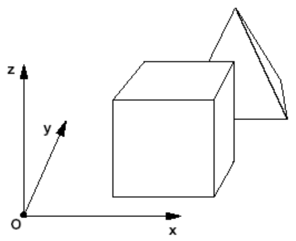


Transformations

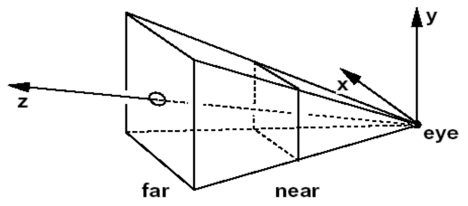
Spaces in CG applications



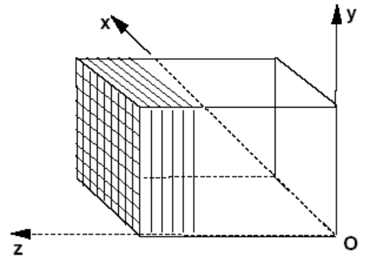
Object Space



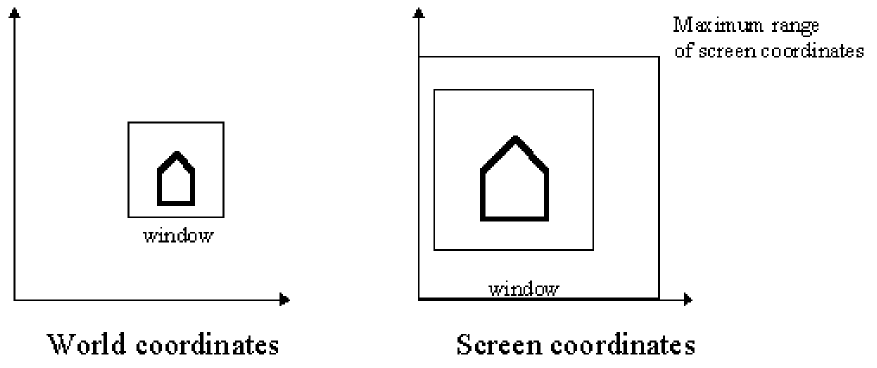
World Space



Camera Space



Viewing Space



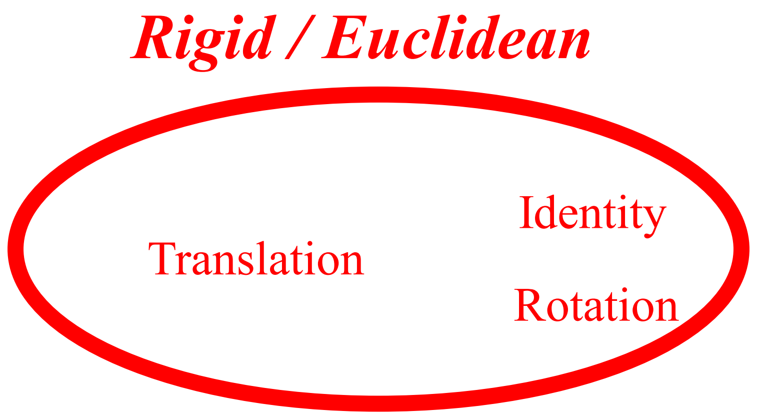
Screen Space

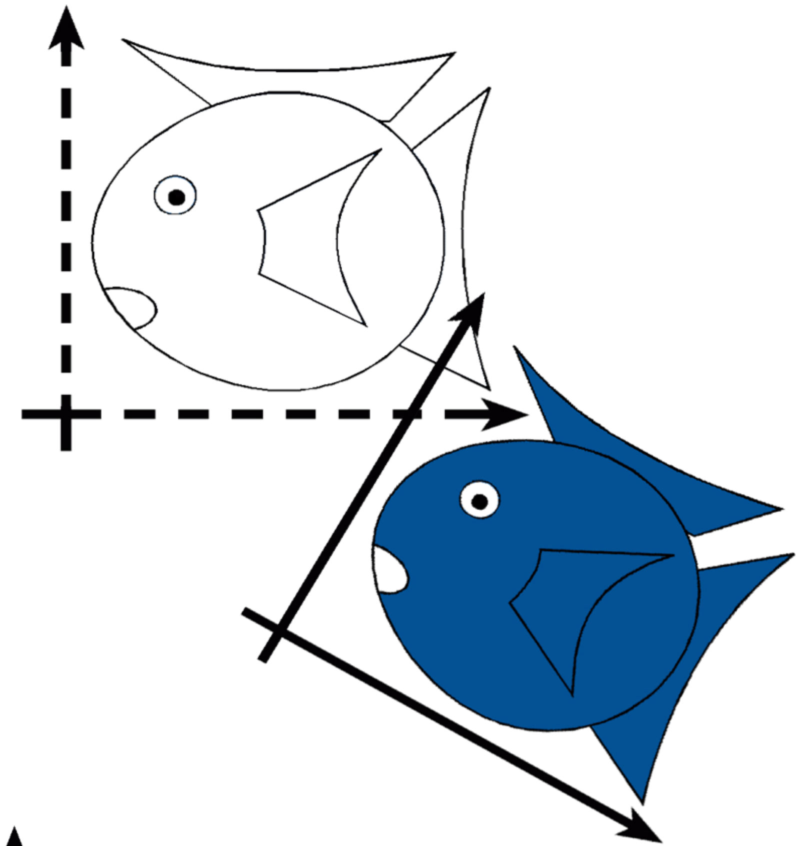
Change spaces using transformations!

Transformations determine what is finally seen (visibility)

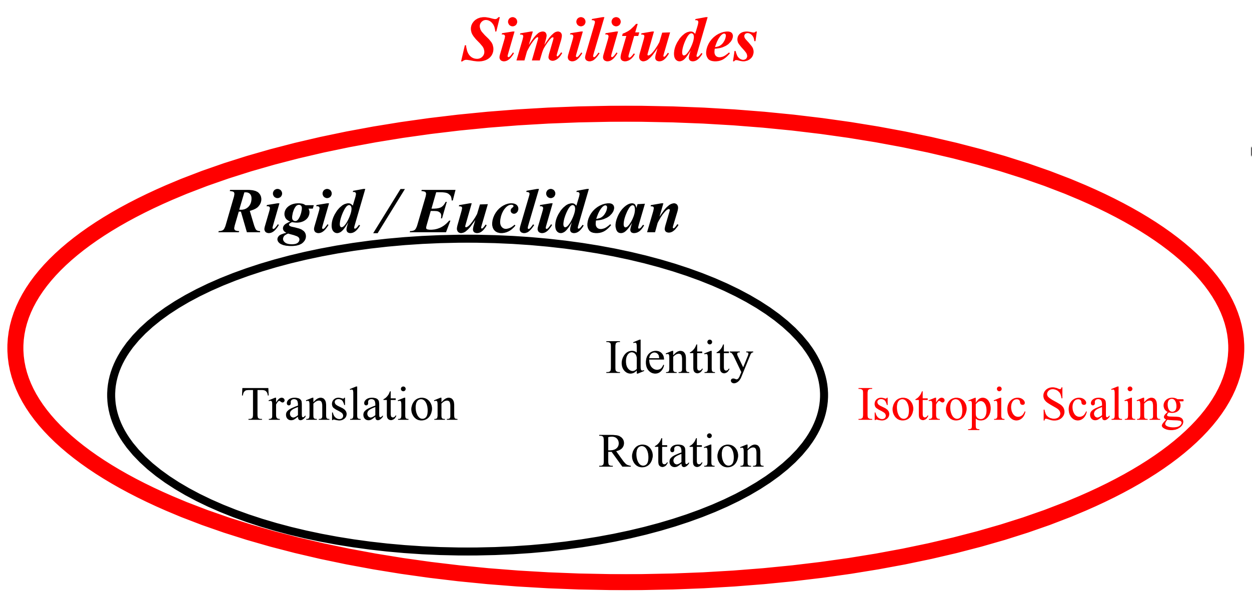
Transformations

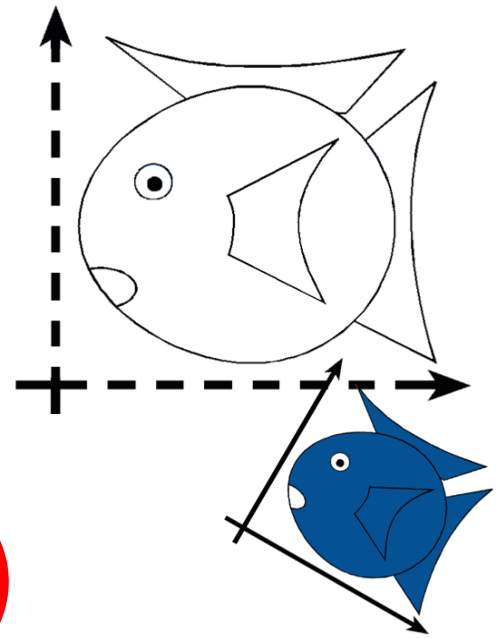
- Rigid Transformations (Euclidean Transform)
  - Preserves distances
  - Preserves angles





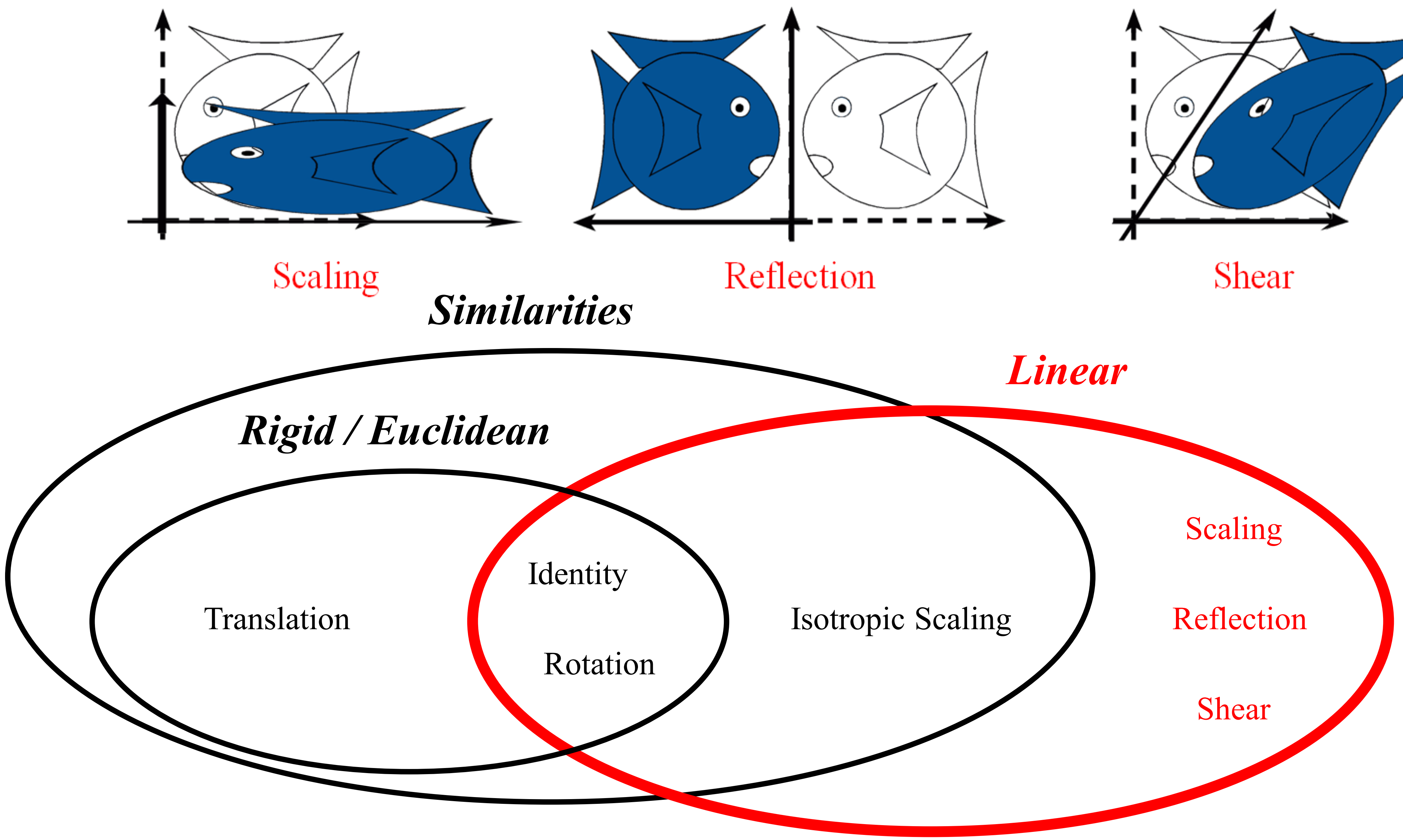
- Similarity Transformations
  - Preserves angles
  - Changes distances





Transformations

Linear Transformations



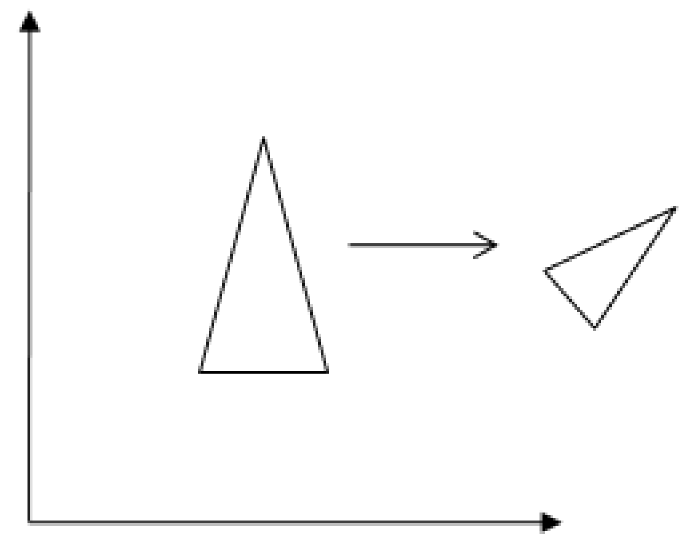
Transformations

Linear Transformations

- Two geometric objects, T_1 and T_2 are congruent if they can be made to coincide by a rigid motion and a positive or negative scaling, i.e. if there exists a scaling factor c <> 0, a translation vector b and an *orthogonal* matrix Q such that

$$T_1 = \mathbf{b} + c\mathbf{Q}T_2$$

T1 und T2 kongruent wenn sie sich umkehrbar lassen
bzw. vom einen in den anderen transformieren lassen



Transformations

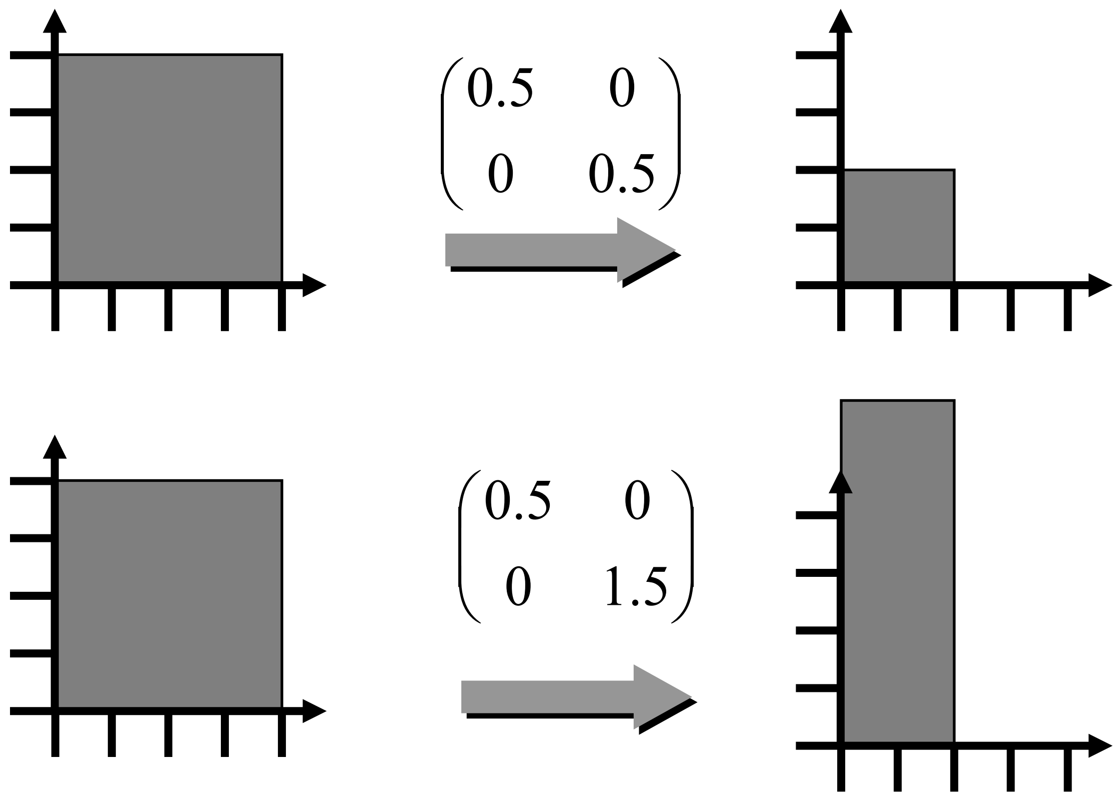
Basic 2D Transformations

- Matrix multiplication
  - Scaling
  - Rotation
- Different operation of transformations on
  - Locations (points)
  - Displacement vectors
  - Normal vectors (surface normals)
- Matrices used for scale, rotation and shear

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a_{11}x + a_{12}y \\ a_{21}x + a_{22}y \end{pmatrix}$$

Transformations

Scaling



- Most basic transformation

$$scale(s_x, s_y) = \begin{pmatrix} s_x & 0 \\ 0 & s_y \end{pmatrix}$$

  - Change length and possibly direction

- Transformation of a vector $\begin{pmatrix} s_x & 0 \\ 0 & s_y \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} s_x x \\ s_y y \end{pmatrix}$

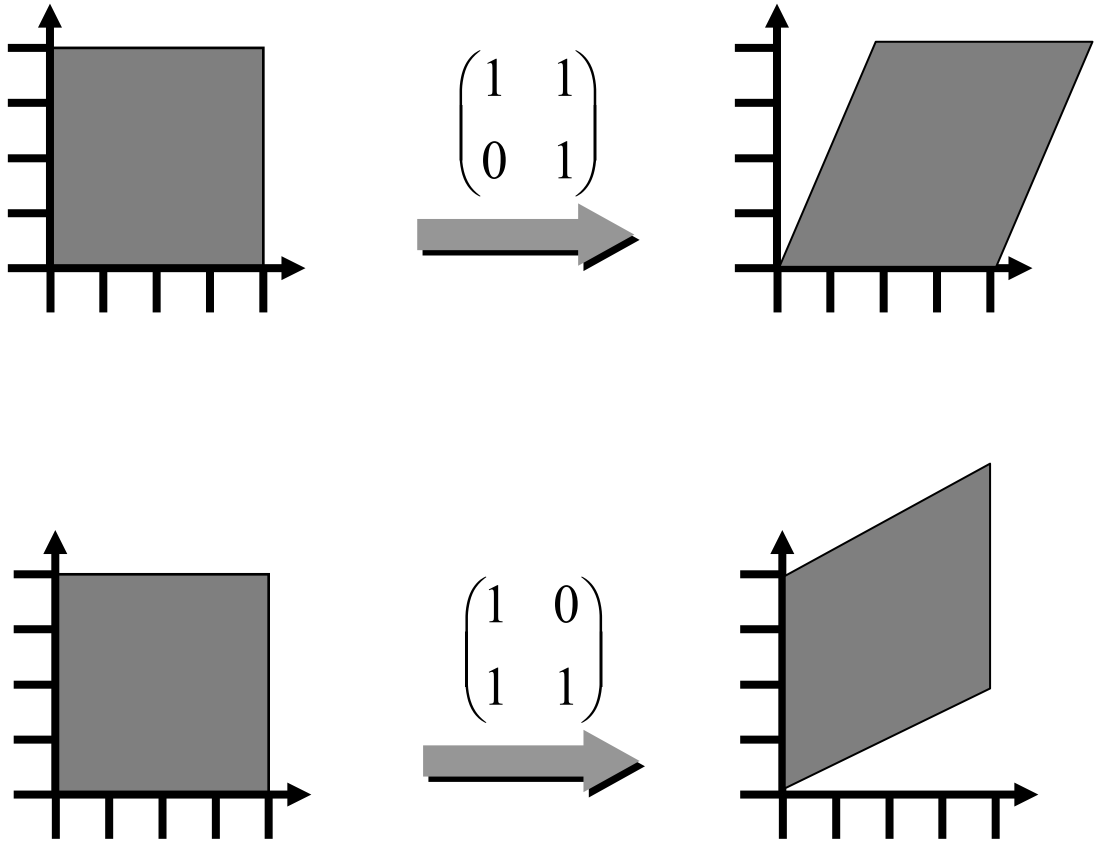
Transformations

Shearing

- Pushing things sideways (deck of cards)
- Horizontal (y-coordinate constant) and Vertical (x-coordinate constant)
- Can be expressed as rotation about an angle

$$\text{shear}-x(s)=\begin{pmatrix}1 & s\\ 0 & 1\end{pmatrix} \qquad \text{shear}-y(s)=\begin{pmatrix}1 & 0\\ s & 1\end{pmatrix}$$

$$\begin{pmatrix}1 & \tan\phi\\ 0 & 1\end{pmatrix} \qquad \begin{pmatrix}1 & 0\\ \tan\phi & 1\end{pmatrix}$$

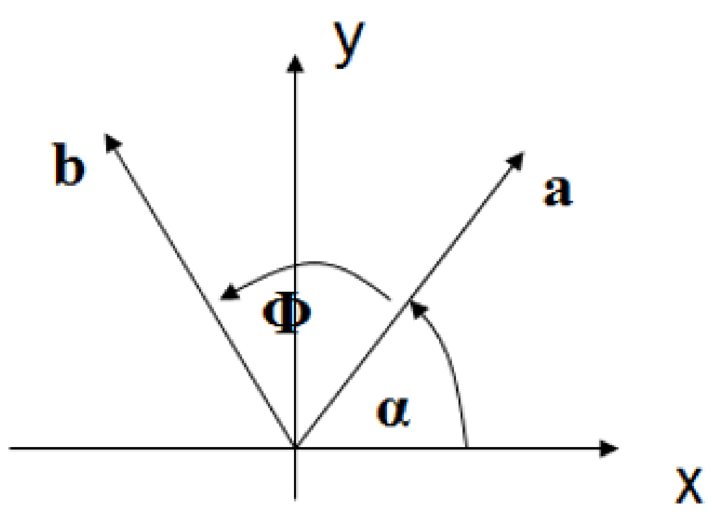


ohm

Transformations

Rotation

- Vector $\mathbf{a} = (a_x, a_y)$
- Length $r = \sqrt{a_x^2 + a_y^2}$
- By definition $a_x = r\cos\alpha$, $a_y = r\sin\alpha$
- Rotation by an angle Φ, counter-clockwise



$$b_x = r\cos(\alpha + \varphi) = r\cos\alpha\cos\varphi - r\sin\alpha\sin\varphi$$

$$b_y = r\sin(\alpha + \varphi) = r\sin\alpha\cos\varphi + r\cos\alpha\sin\varphi$$

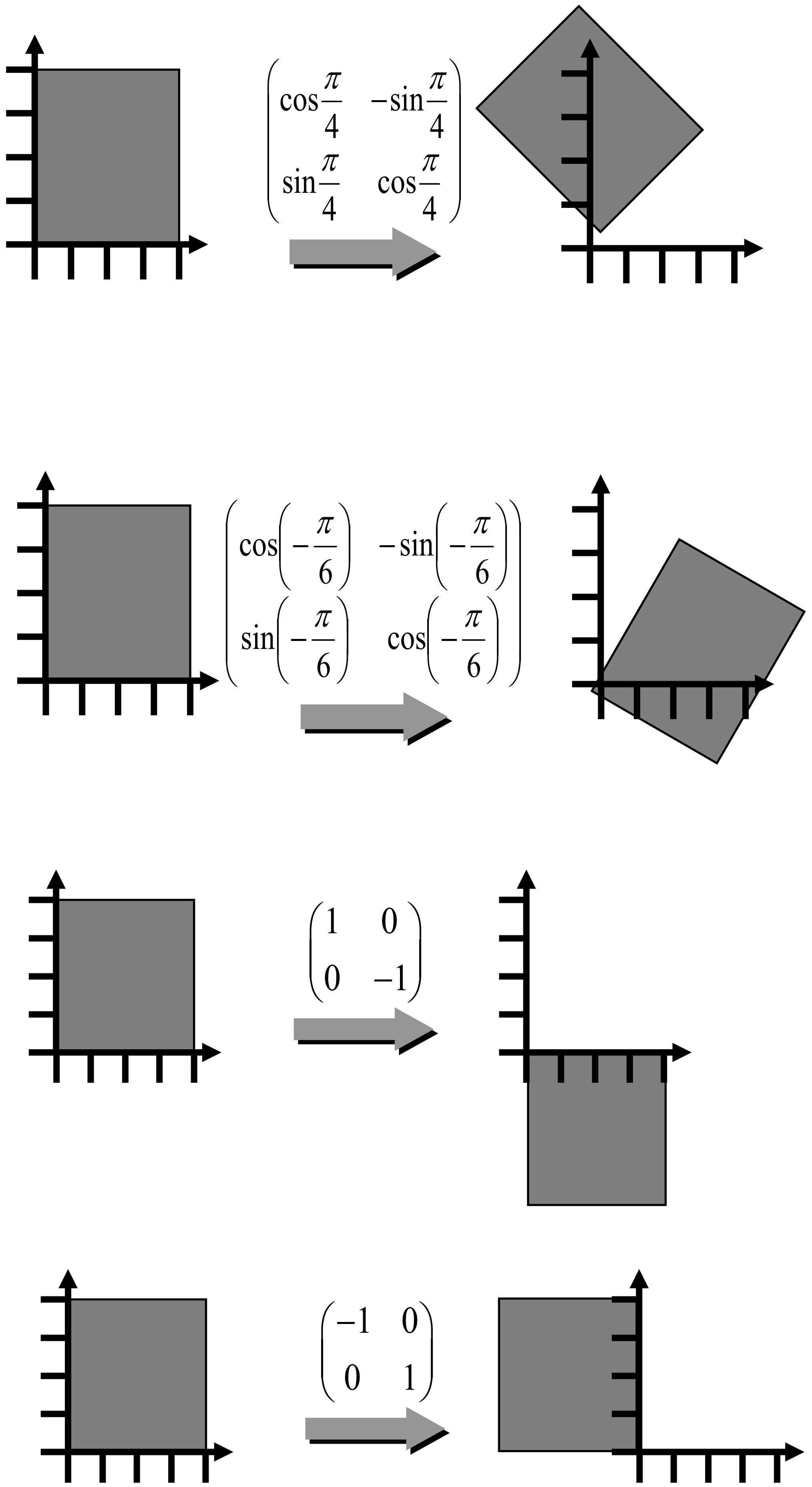
Transformations

- Substitute

$$b_x = a_x \cos\varphi - a_y \sin\varphi$$
$$b_y = a_y \cos\varphi + a_x \sin\varphi$$

- Matrix from a to b

$$rotate(\phi) = R(\phi) = \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix}$$



ohm

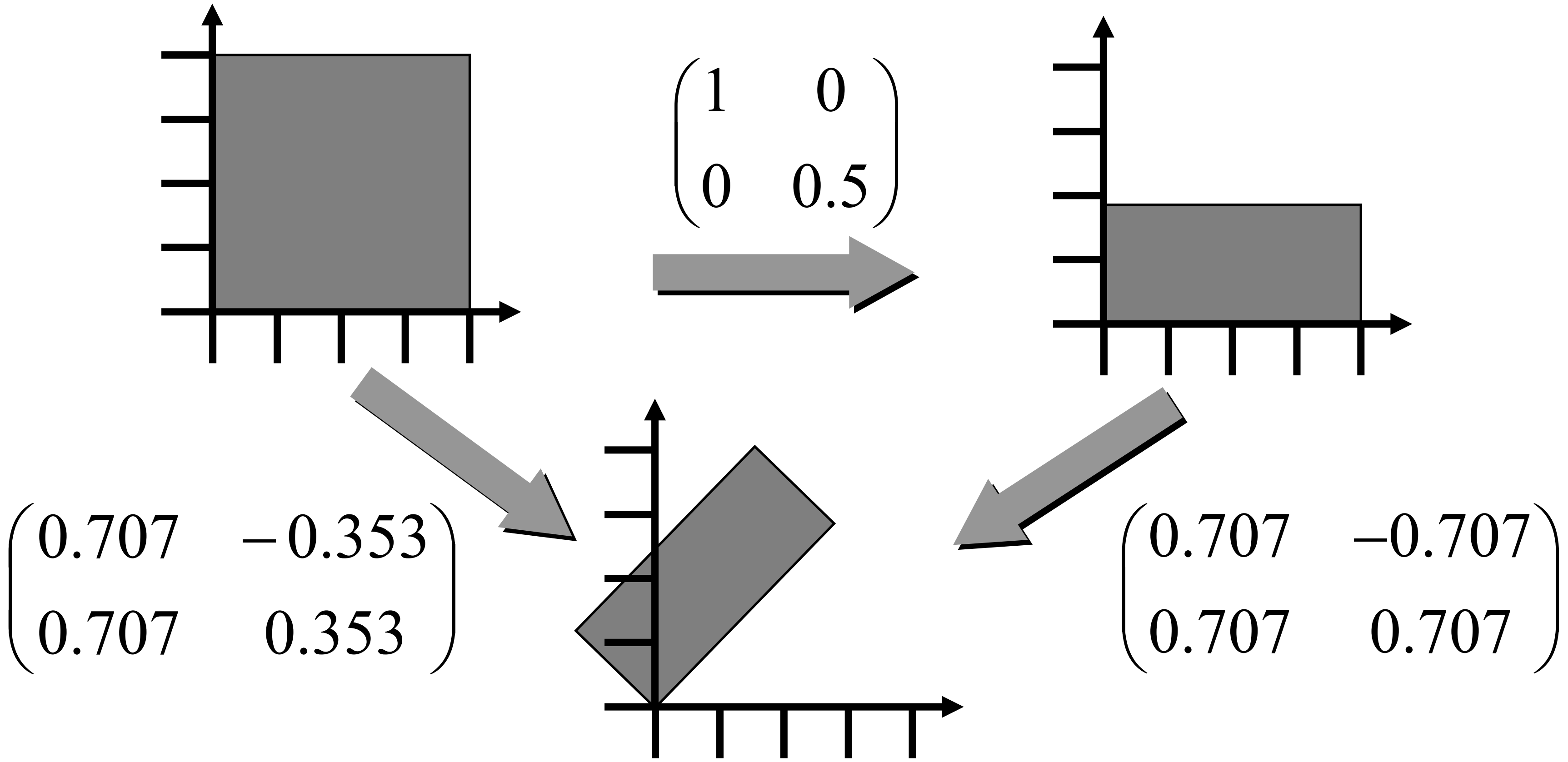
Transformations

Composition of 2D Transformations

- First $\mathbf{v}_2 = \mathbf{S}\mathbf{v}_1$
- Second $\mathbf{v}_3 = \mathbf{R}\mathbf{v}_2$

2. 1.

or $\mathbf{v}_3 = \mathbf{R}(\mathbf{S}\mathbf{v}_1)$



Transformations

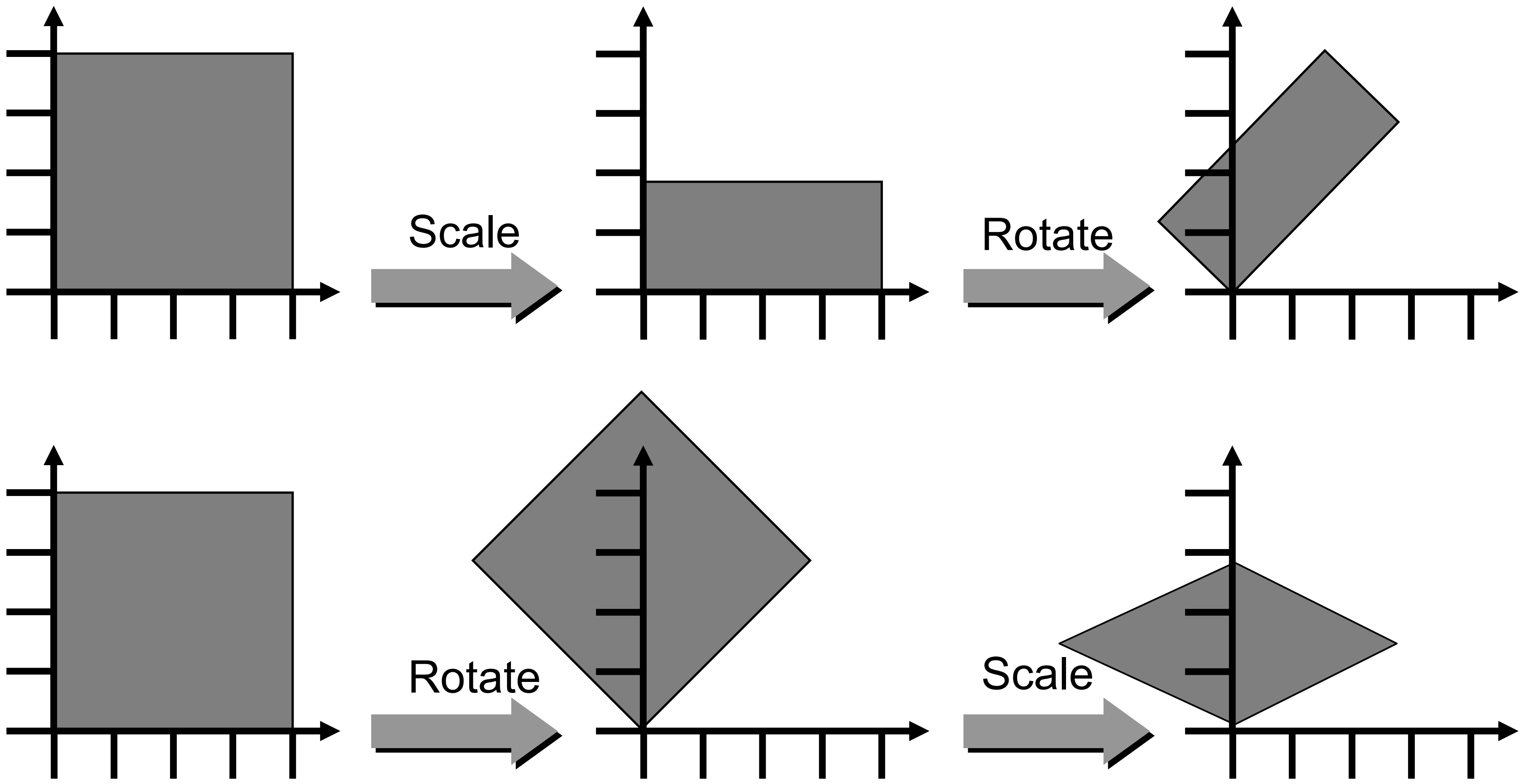
- Matrix multiplications are associative (R S) T = R (S T)

$$\Rightarrow \quad \mathbf{v}_3 = (\mathbf{RS})\mathbf{v}_1 = \mathbf{M}\mathbf{v}_1$$

  - Effects of two consecutive matrix multiplications by one matrix
  - Applied **from the right side** first! Here: first apply S, then R
- Matrix multiplications are *NOT commutative*
  - **The order of transformations does matter**
  - Note the difference
    - Scaling, then rotating vs. rotating, then scaling

Transformations

Order does matter



ohm

Transformations

3D Transformations

- Rotation about the main xyz-axis
  - Same applies to scale / shear

$$Rot_x(\phi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{pmatrix}$$

$$Rot_y(\phi) = \begin{pmatrix} \cos\phi & 0 & \sin\phi \\ 0 & 1 & 0 \\ -\sin\phi & 0 & \cos\phi \end{pmatrix}$$

$$Rot_z(\phi) = \begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Transformations

Arbitrary 3D rotations

- 3D rotations by *orthogonal* matrices that preserve orientation
- A matrix is orthogonal if

$$O^T \cdot O = O \cdot O^T = I_{nxn} \text{ and } \det(O) = 1$$

orthogonal => inverse == transponierte

- Matrix rows
  - Cartesian coordinates of three **mutually orthogonal** unit vectors
- Matrix columns
  - Three potentially different **mutually orthogonal** unit vectors

Transformations

Change of base

- Let $\vec{u}, \vec{v}, \vec{w}$ form an orthonormal system

$$\vec{u} \cdot \vec{u} = \vec{v} \cdot \vec{v} = \vec{w} \cdot \vec{w} = 1$$

$$\vec{u} \cdot \vec{v} = \vec{v} \cdot \vec{w} = \vec{w} \cdot \vec{u} = 0$$

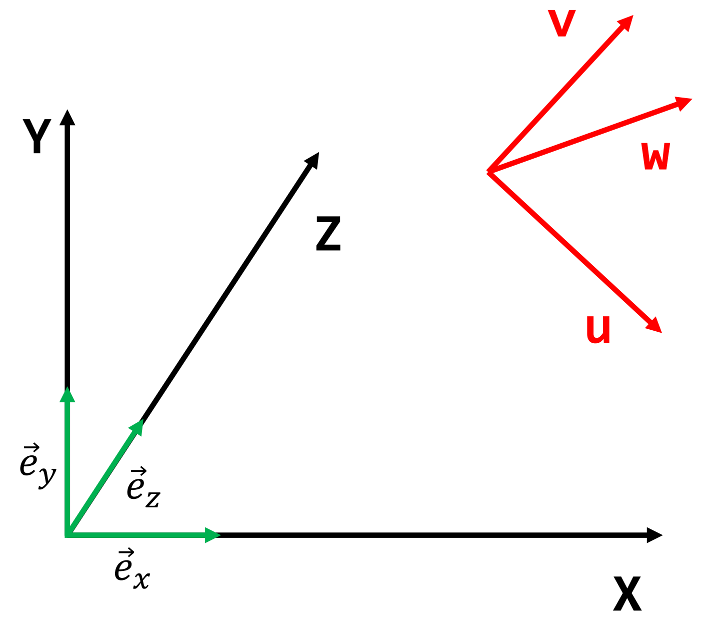
- with

$$\vec{u} = u_x \vec{e}_x + u_y \vec{e}_y + u_z \vec{e}_z$$

$$\vec{v} = v_x \vec{e}_x + v_y \vec{e}_y + v_z \vec{e}_z$$

$$\vec{w} = w_x \vec{e}_x + w_y \vec{e}_y + w_z \vec{e}_z$$

- then the associated rotation matrix is $\boldsymbol{R}_{uvw} = \begin{pmatrix} u_x & u_y & u_z \\ v_x & v_y & v_z \\ w_x & w_y & w_z \end{pmatrix}$



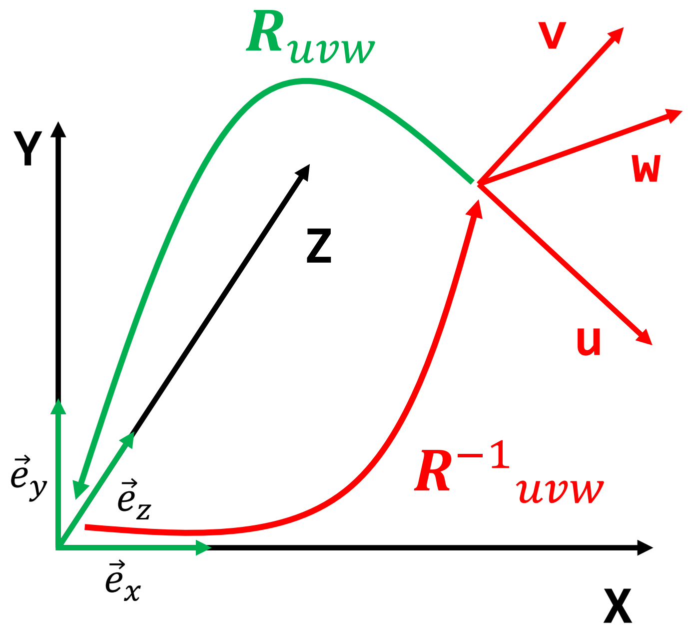
Transformations

- The matrix $\boldsymbol{R}_{uvw}$
  - Takes the basis uvw to the corresponding cartesian axis via rotation

$$\boldsymbol{R}_{uvw}\vec{u} = \begin{pmatrix} u_x & u_y & u_z \\ v_x & v_y & v_z \\ w_x & w_y & w_z \end{pmatrix}\begin{pmatrix} u_x \\ u_y \\ u_z \end{pmatrix} = \begin{pmatrix} \vec{u} \cdot \vec{u} \\ \vec{v} \cdot \vec{u} \\ \vec{w} \cdot \vec{u} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \vec{e}_x$$

$$\boldsymbol{R}_{uvw}\vec{v} = \vec{e}_y$$

$$\boldsymbol{R}_{uvw}\vec{w} = \vec{e}_z$$

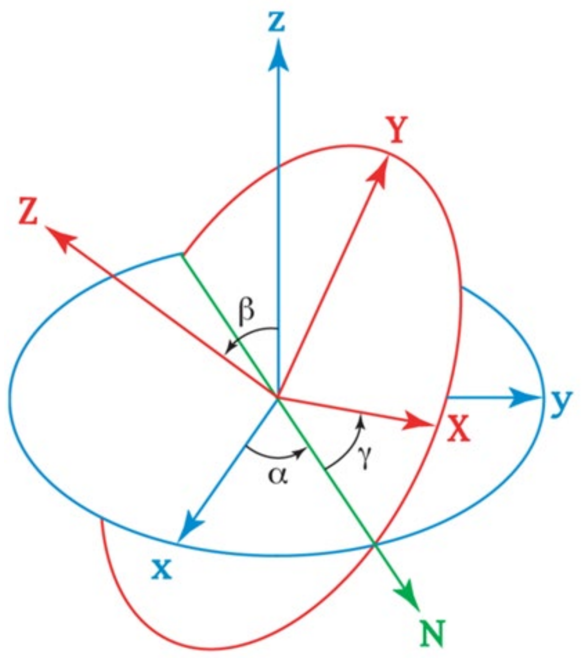


  - If $\boldsymbol{R}_{uvw}$ is a rotation matrix with orthnormal rows then $\boldsymbol{R}^T{}_{uvw}$ is a rotation matrix with orthnormal columns and $\Rightarrow \boldsymbol{R}^T{}_{uvw} = \boldsymbol{R}^{-1}{}_{uvw}$
  - Consequence: change coordinate systems by using the basis vectors of an orthonormal system as columns of a rotation matrix

ohm

Transformations

Equivalent representations of rotations in 3D



- Orthogonal matrices
- 3 Euler rotations

$$\mathbf{R} = \mathbf{R}_z(\alpha)\cdot\mathbf{R}_y(\beta)\cdot\mathbf{R}_x(\gamma) = \begin{pmatrix} \cos\beta\cos\alpha & \sin\gamma\sin\beta\cos\alpha - \cos\gamma\sin\alpha & \cos\gamma\sin\beta\cos\alpha + \sin\gamma\sin\alpha \\ \cos\beta\sin\alpha & \sin\gamma\sin\beta\sin\alpha + \cos\gamma\cos\alpha & \cos\gamma\sin\beta\sin\alpha - \sin\gamma\cos\alpha \\ -\sin\beta & \sin\gamma\cos\beta & \cos\gamma\cos\beta \end{pmatrix}$$

- Axis of rotation and angle
- Quaternions
- 2 (planar) reflections

Transformations

For orthogonal transformations

- Algebraic inverse = geometric inverse = transpose
- So, if R_{uvw} takes v to y then R^T_{uvw} takes y to v
- Construct rotation about arbitrary vector a
  - Form orthonormal basis with w = a
  - Rotate basis to canonical basis xyz
  - Rotate about z-axis
  - Rotate canonical basis back to uvw basis

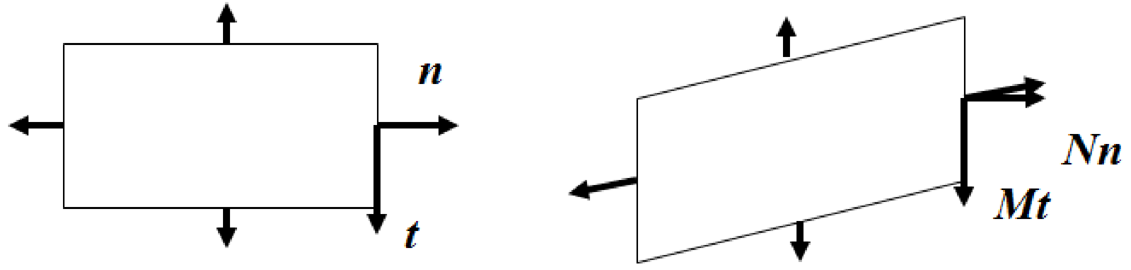
$$\begin{pmatrix} u_x & v_x & w_x \\ u_y & v_y & w_y \\ u_z & v_z & w_z \end{pmatrix}\begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} u_x & u_y & u_z \\ v_x & v_y & v_z \\ w_x & w_y & w_z \end{pmatrix}$$

ohm

Transformations

Transformation of surface normal vectors

- Problem: Transformation of surface normals differs from transformation of underlying surface:



  - We have: $\mathbf{n}^T \cdot \mathbf{t} = 0$ and $\mathbf{t}_M = \mathbf{M}\mathbf{t}$ *and* $\mathbf{n}_N = \mathbf{N}\mathbf{n}$
  - Goal: find N, such that $\mathbf{n}_N^T \cdot \mathbf{t}_M = 0$
  - And $\mathbf{M}^{-1}\mathbf{M} = \mathbf{I}$, hence $\Rightarrow \mathbf{n}_M^T = \mathbf{n}^T\mathbf{M}^{-1}$

$$\mathbf{n}^T \cdot \mathbf{t} = \mathbf{n}^T\mathbf{I}\mathbf{t} = \mathbf{n}^T\mathbf{M}^{-1}\mathbf{M}\mathbf{t} = 0$$

$$\left(\mathbf{n}^T\mathbf{M}^{-1}\right)\cdot\left(\mathbf{M}\mathbf{t}\right) = \left(\mathbf{n}^T\mathbf{M}^{-1}\right)\cdot \mathbf{t}_M = 0$$

$$\mathbf{n}_M = \left(\mathbf{n}^T\mathbf{M}^{-1}\right)^T = \left(\mathbf{M}^{-1}\right)^T \mathbf{n}$$

$$\Rightarrow \boxed{\mathbf{N} = \left(\mathbf{M}^{-1}\right)^T}$$

Transform with the inverse transpose of the original transformation matrix!

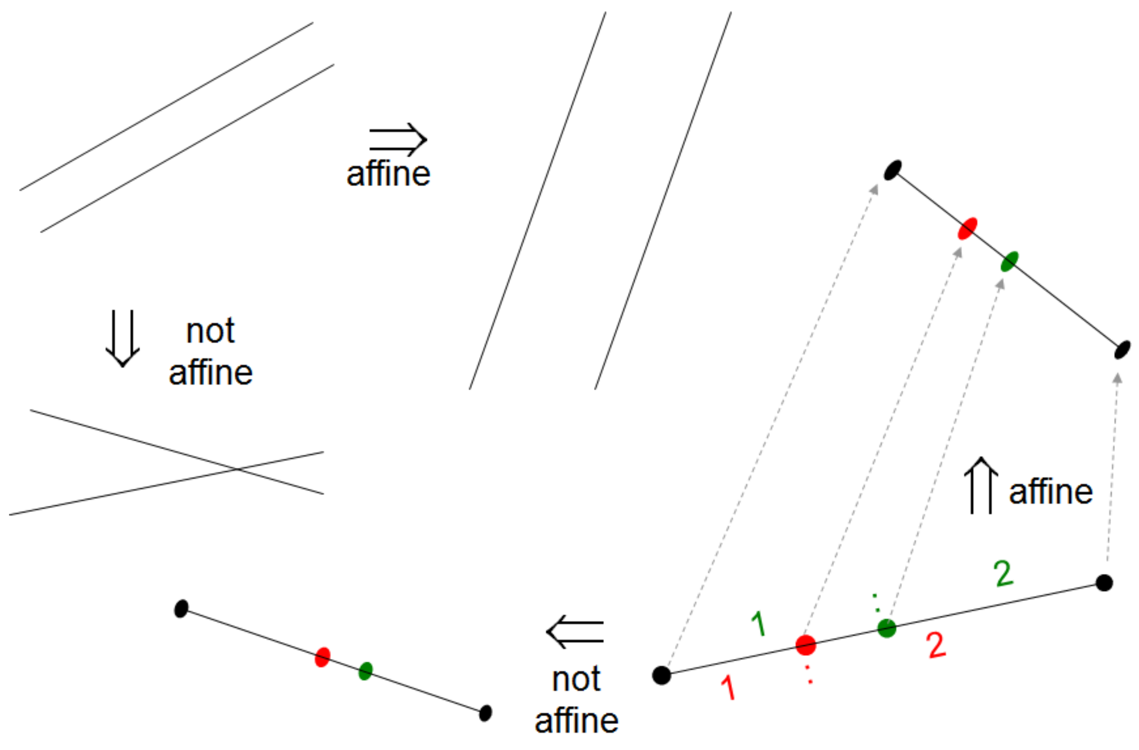
ohm

Affine Transformations

- Linear Transformation and Translation

$$\vec{x} \mapsto A\vec{x} + \vec{b} \quad = \quad \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

- Characterization
  - Maps lines to lines
  - Parallel lines will be mapped to parallel lines
  - Division rations will be kept
  - Angles are not preserved



- Representation in CG: **homogenous coordinates**

Affine Transformations

- Dilemma 1
  - Vector $\vec{v} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$: describes a direction in space – independent of position
  - You cannot "translate" direction vectors!
- Dilemma 2

$$T = T_1 T_2 T_3 \ldots$$

  - Linear transformations can be simply concatenated by matrix multiplication

$$T_1(\vec{x}) = \boldsymbol{M}_1\vec{x} + \vec{t}_1 \qquad T_2(\vec{x}) = \boldsymbol{M}_2\vec{x} + \vec{t}_2$$

  - Affine Transformations

$$\Rightarrow T_2\left(T_1(\vec{x})\right) = \boldsymbol{M}_2\boldsymbol{M}_1\vec{x} + \boldsymbol{M}_2\vec{t}_1 + \vec{t}_2$$

ohm

Affine Transformations

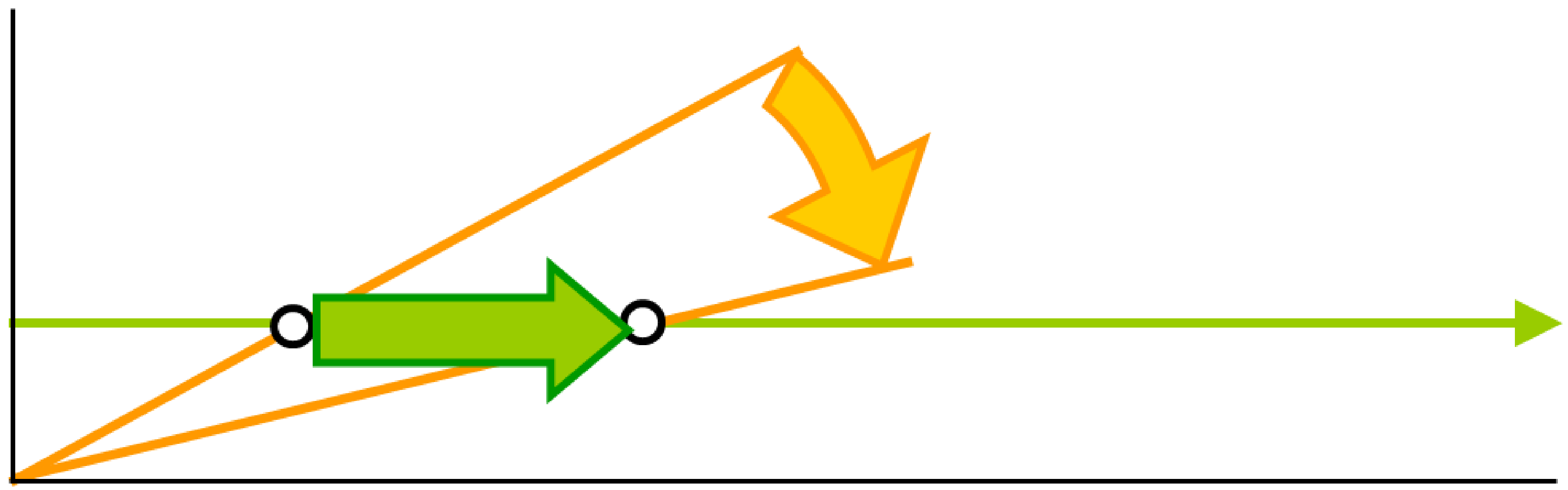
Homogenous coordinates (simple)

- Add third / fourth (2d/3d) coordinate w (1 for point, 0 for vector)
- Homogenous coordinates (advanced)
  - Identify (x,y) with the line $\{\alpha x, \alpha y, \alpha | \alpha \in \mathbb{R}\}$ in 3D or with any non-zero point on this line (e.g. (x,y,1))
  - Consequence: $(x, y, 1)$, $(3x, 3x, 3)$, $(0.5x, 0.5y, 0.5)$ represent the same point!
  - Dehomogenization: $(x, y, w) \Rightarrow (\frac{x}{w}, \frac{y}{w})$
  - Analogous in 3D
- Mathematical foundation
  - Projective geometry

$$\vec{p}_H = \begin{pmatrix} x \\ y \\ z \\ w \end{pmatrix} \leftrightarrow \vec{p} = \begin{pmatrix} \frac{x}{w} \\ \frac{y}{w} \\ \frac{z}{w} \end{pmatrix}$$

Affine Transformations

- Example: Point in 1D
  - Corresponds to ray in homogenous coordinates
- Rotation in homogenous coordinates
  - Corresponds to translation in cartesian coordinates



ohm

Affine Transformations

- What is w?
  - For points: $w = 1$
  - For vectors: $w = 0$
  - For matrices
    - Add row $(0 \quad 0 \quad 0 \quad 1)$
    - Rest of remaining column contains $\vec{t}$
- Result

$$\vec{p} = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \qquad \vec{v} = \begin{pmatrix} x \\ y \\ z \\ 0 \end{pmatrix} \qquad \boldsymbol{M} = \begin{pmatrix} m_{00} & m_{01} & m_{02} & t_x \\ m_{10} & m_{11} & m_{12} & t_y \\ m_{20} & m_{21} & m_{22} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

  - Note: homogenous transformation matrices can also be used for projection transformations

Affine Transformations

- Homogenous Coordinates: General form in 3D

$$\vec{x} \mapsto \mathbf{M}\vec{x} + \vec{b} \Rightarrow$$

$$= \begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

$$\begin{array}{cc} & \begin{array}{cc} \text{lin. Transf.} & \text{Translation} \end{array} \\ = & \left(\begin{array}{ccc|c} m_{11} & m_{12} & m_{13} & b_1 \\ m_{21} & m_{22} & m_{23} & b_2 \\ m_{31} & m_{32} & m_{33} & b_3 \\ 0 & 0 & 0 & 1 \end{array}\right) \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \end{array}$$

ohm

Affine Transformations

- Affine transformation of point $\vec{p}$

$$\vec{p}' = \boldsymbol{M}\vec{p} = \begin{pmatrix} m_{00} & m_{01} & m_{02} & t_x \\ m_{10} & m_{11} & m_{12} & t_y \\ m_{20} & m_{21} & m_{22} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} m_{00}x + m_{01}y + m_{02}z + t_x \cdot 1 \\ m_{10}x + m_{11}y + m_{12}z + t_y \cdot 1 \\ m_{20}x + m_{21}y + m_{22}z + t_z \cdot 1 \\ 0 \cdot x + 0 \cdot y + 0 \cdot z + 1 \end{pmatrix} = \begin{pmatrix} x' + t_x \\ y' + t_y \\ z' + t_z \\ 1 \end{pmatrix}$$

- Affine transformation of vector $\vec{v}$

$$\vec{v}' = \boldsymbol{M}\vec{v} = \begin{pmatrix} m_{00} & m_{01} & m_{02} & t_x \\ m_{10} & m_{11} & m_{12} & t_y \\ m_{20} & m_{21} & m_{22} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 0 \end{pmatrix} = \begin{pmatrix} m_{00}x + m_{01}y + m_{02}z + t_x \cdot 0 \\ m_{10}x + m_{11}y + m_{12}z + t_y \cdot 0 \\ m_{20}x + m_{21}y + m_{22}z + t_z \cdot 0 \\ 0 \cdot x + 0 \cdot y + 0 \cdot z + 1 \cdot 0 \end{pmatrix} = \begin{pmatrix} x' \\ y' \\ z' \\ 0 \end{pmatrix}$$

- Note how direction vectors are not translated!

Affine Transformations

Concatenation of transformations

- Regular (cartesian coordinates)

$$T_1(\vec{x}) = \boldsymbol{M}_1\vec{x} + \vec{t}_1 \qquad T_2(\vec{x}) = \boldsymbol{M}_2\vec{x} + \vec{t}_2$$

$$\Rightarrow T_2\left(T_1(\vec{x})\right) = \boldsymbol{M}_2\boldsymbol{M}_1\vec{x} + \boldsymbol{M}_2\vec{t}_1 + \vec{t}_2$$

- Homogenous coordinates

$$T_1(\vec{x}) = \boldsymbol{M}_1\vec{x} \qquad T_2(\vec{x}) = \boldsymbol{M}_2\vec{x}$$

$$\Rightarrow T_2\left(T_1(\vec{x})\right) = \boldsymbol{M}_2\boldsymbol{M}_1\vec{x}$$

Affine Transformations

Transformation rules

- Multiplication is composition

$$x \overset{T}{\mapsto} Tx = y \overset{S}{\mapsto} Sy = z \quad \equiv \quad z = A_S \cdot A_T \cdot x$$

- *Inverse matrix* will inverse the transformation (inverse transformation)
  - Affine transformations are invertible
  - Caution: projection transformations are not generally invertible!
    - But also expressed as a homogenous matrix

ohm

Affine Transformations

Common transformations

- Translation and scaling

$$\boldsymbol{T} = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad \boldsymbol{S}(a,b,c) = \begin{pmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Rotation around the x-, y- and z-axis (note: φ is in radians)

$$\boldsymbol{R_x}(\varphi) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi & 0 \\ 0 & \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad \boldsymbol{R_y}(\varphi) = \begin{pmatrix} \cos\varphi & 0 & \sin\varphi & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\varphi & 0 & \cos\varphi & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad \boldsymbol{R_z}(\varphi) = \begin{pmatrix} \cos\varphi & -\sin\varphi & 0 & 0 \\ \sin\varphi & \cos\varphi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Affine Transformations

Example of a general transformation in 2D

- Rotate with center at c=(cx,cy) by angle ϕ
- Transl(-c) $\rightarrow$ Rot(ϕ) $\rightarrow$ Transl(c)

- in den Ursprung transformieren um -c
- rotieren
- zurück verschieben auf c

$$\begin{pmatrix} 1 & 0 & c_x \\ 0 & 1 & c_y \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & -c_x \\ 0 & 1 & -c_y \\ 0 & 0 & 1 \end{pmatrix}$$

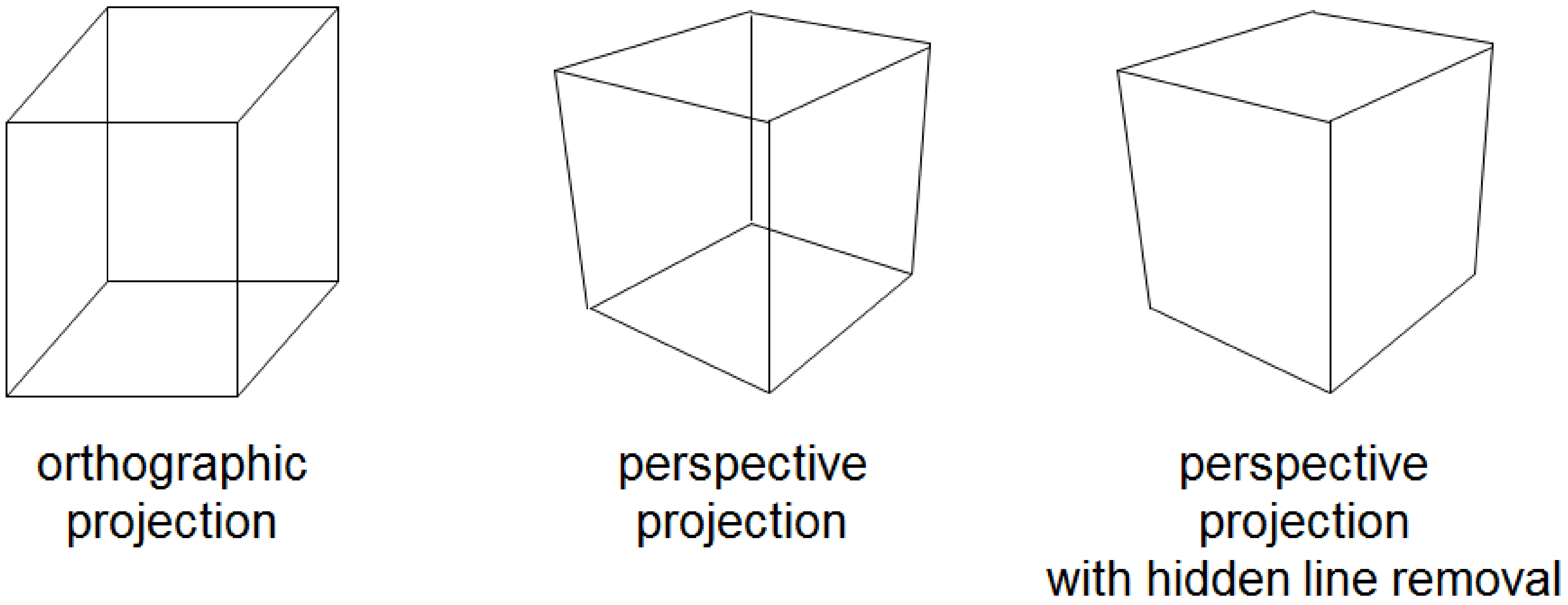
$$\Rightarrow \begin{pmatrix} \cos\varphi & -\sin\varphi & (c_y \sin\varphi - c_x \cos\varphi) + c_x \\ \sin\varphi & \cos\varphi & (-c_x \sin\varphi - c_y \cos\varphi) + c_y \\ 0 & 0 & 1 \end{pmatrix}$$

Projections

Projections

Viewing

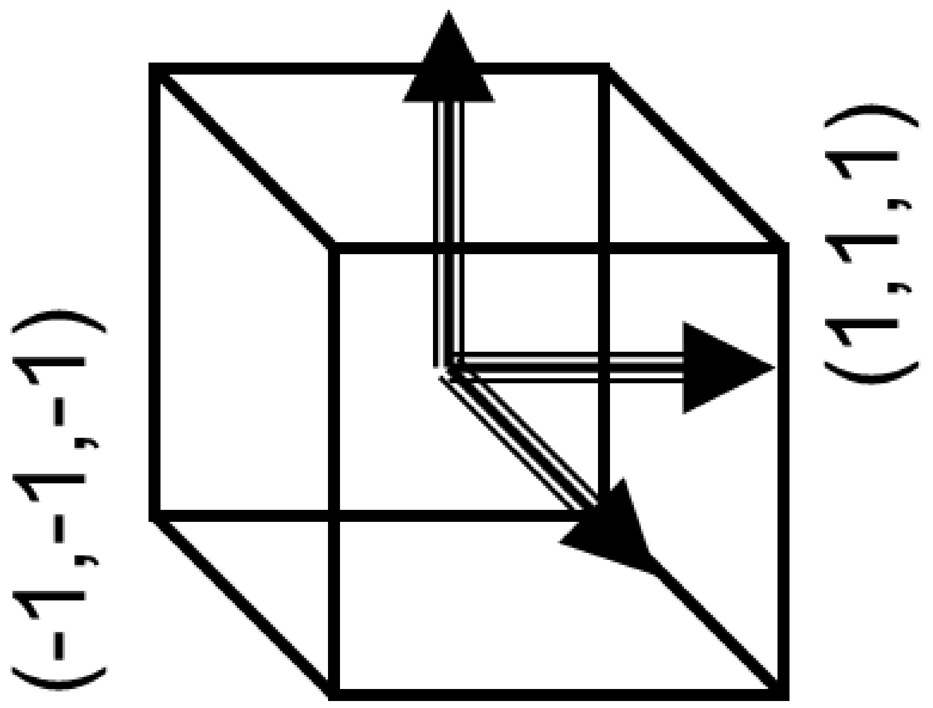
- Orthographic projection: parallel lines map to parallel lines
- Perspective projection: have 1, 2 or 3 vanishing points



Projections

The canonical view volume

- Basic problem: map lines to the screen
  - Reuse solution for any viewing condition
- Limit to canonical view volume (x,y,z) in $[-1, 1]^3$
- If screen has nx x ny pixels
  - $x = -1 \Rightarrow$ left side
  - $x = +1 \Rightarrow$ right side
  - $y = +1 \Rightarrow$ top
  - $y = -1 \Rightarrow$ bottom
- Note the mapping: "square" $\Rightarrow$ rectangle
- *Projections* transform the scene into the canonical view volume

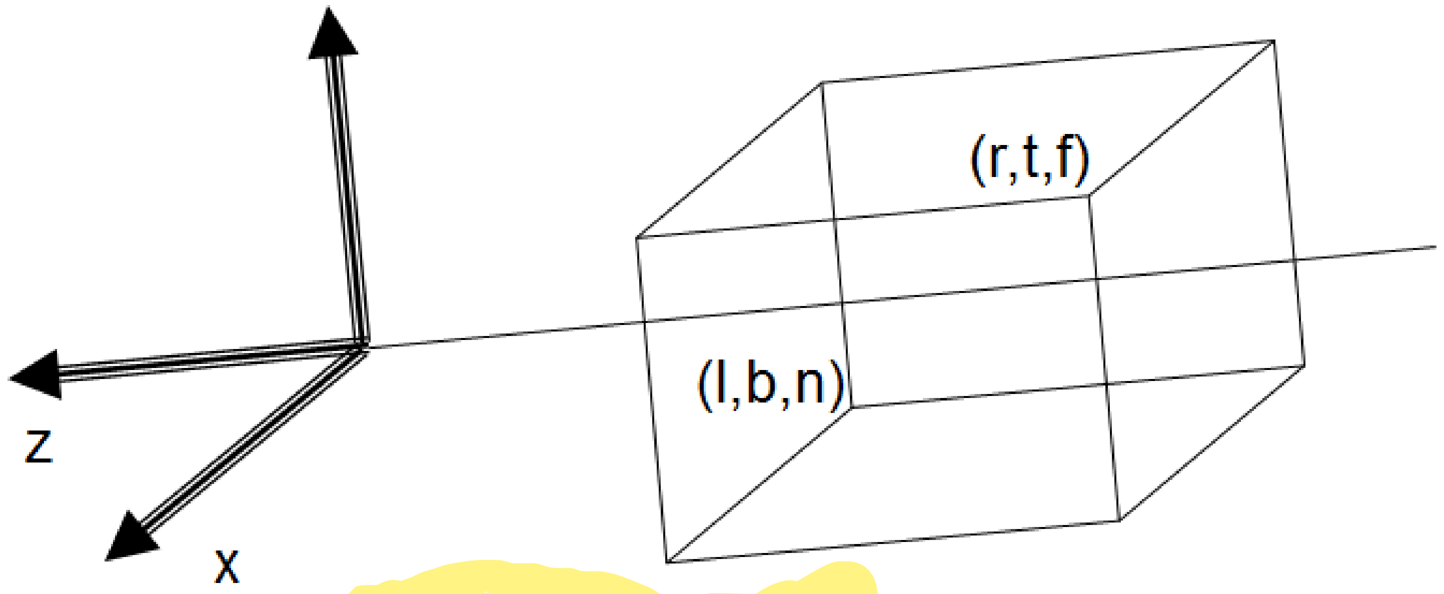


ohm

Projections

Orthographic projection

- Viewer looks in direction of negative z-axis
- y-direction: view up
- x-axis: pointing right (right-handed coordinate system)



$$M_{ortho} = \begin{pmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & -\frac{2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$Note \quad \Rightarrow \quad n > f$

Projections

Orthographic projection dissected

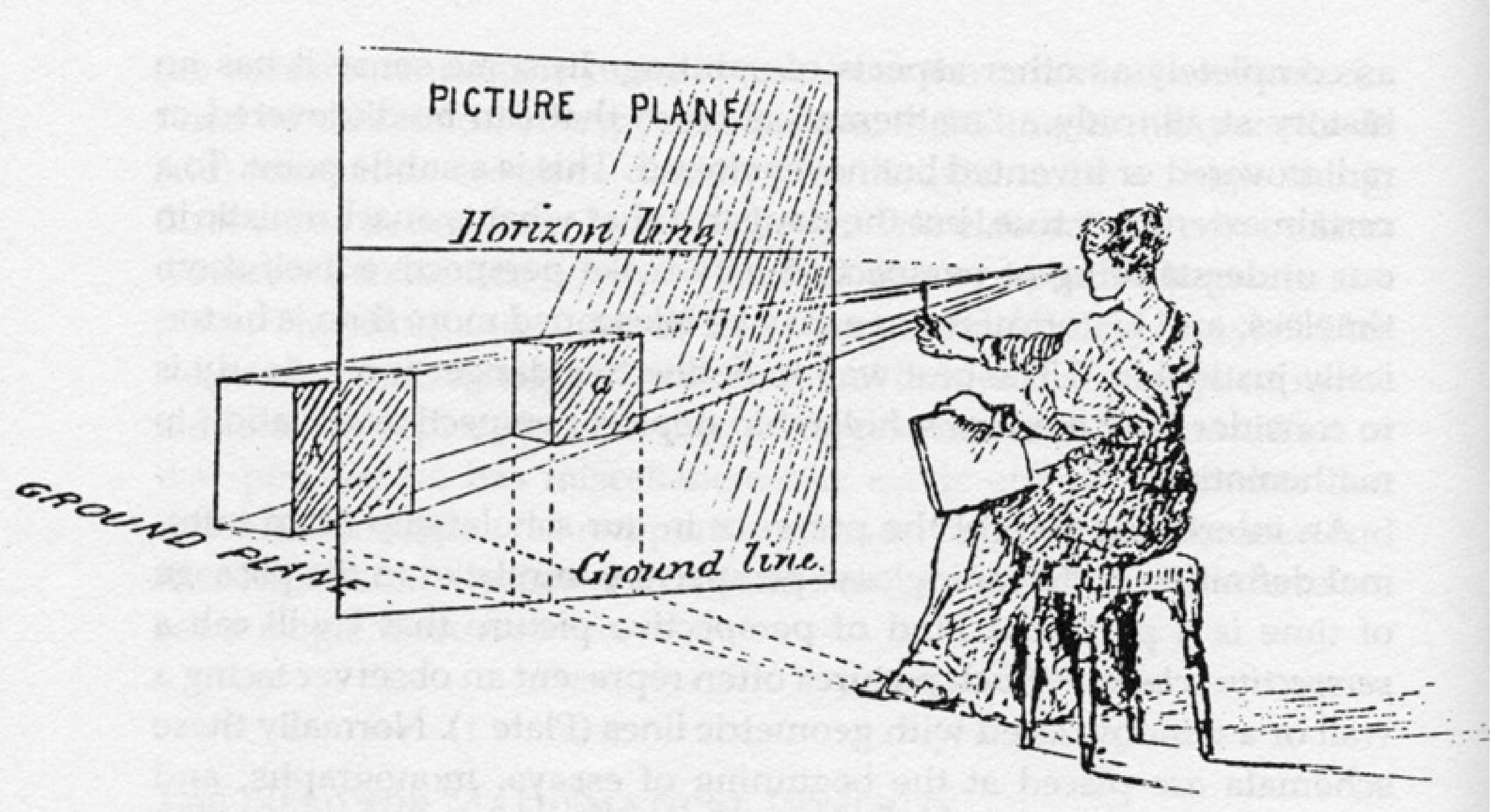
- First: move to the origin
- Second: scale to match canonical view volume

$$\begin{pmatrix} x_{canonical} \\ y_{canonical} \\ z_{canonical} \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} \frac{2}{r-l} & 0 & 0 & 0 \\ 0 & \frac{2}{t-b} & 0 & 0 \\ 0 & 0 & \frac{2}{n-f} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{\text{Scale}} \cdot \underbrace{\begin{pmatrix} 1 & 0 & 0 & -\frac{l+r}{2} \\ 0 & 1 & 0 & -\frac{b+t}{2} \\ 0 & 0 & 1 & -\frac{n+f}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{\text{Translation}} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

ohm

Projections

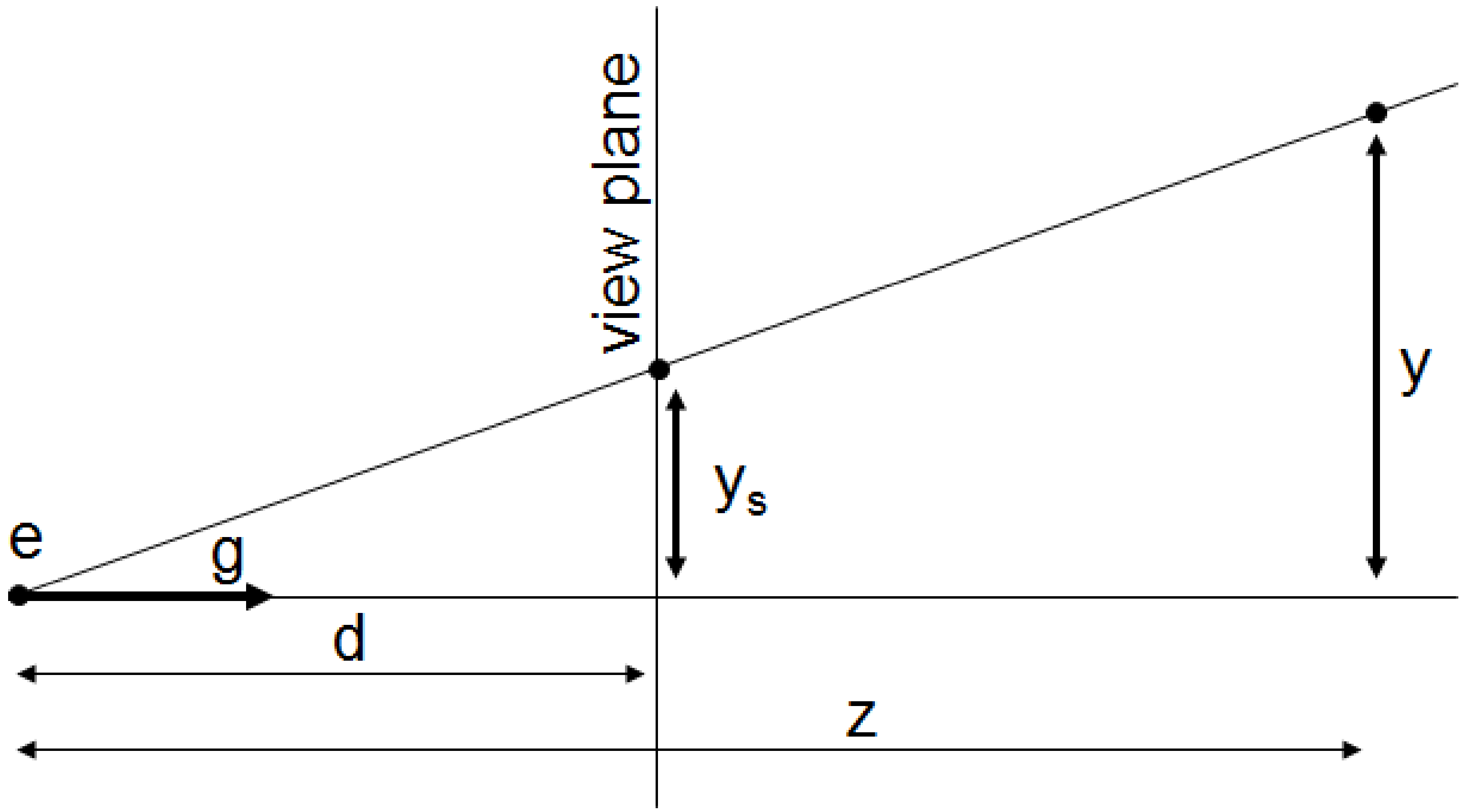
Perspective projection



Projections

Perspective projection

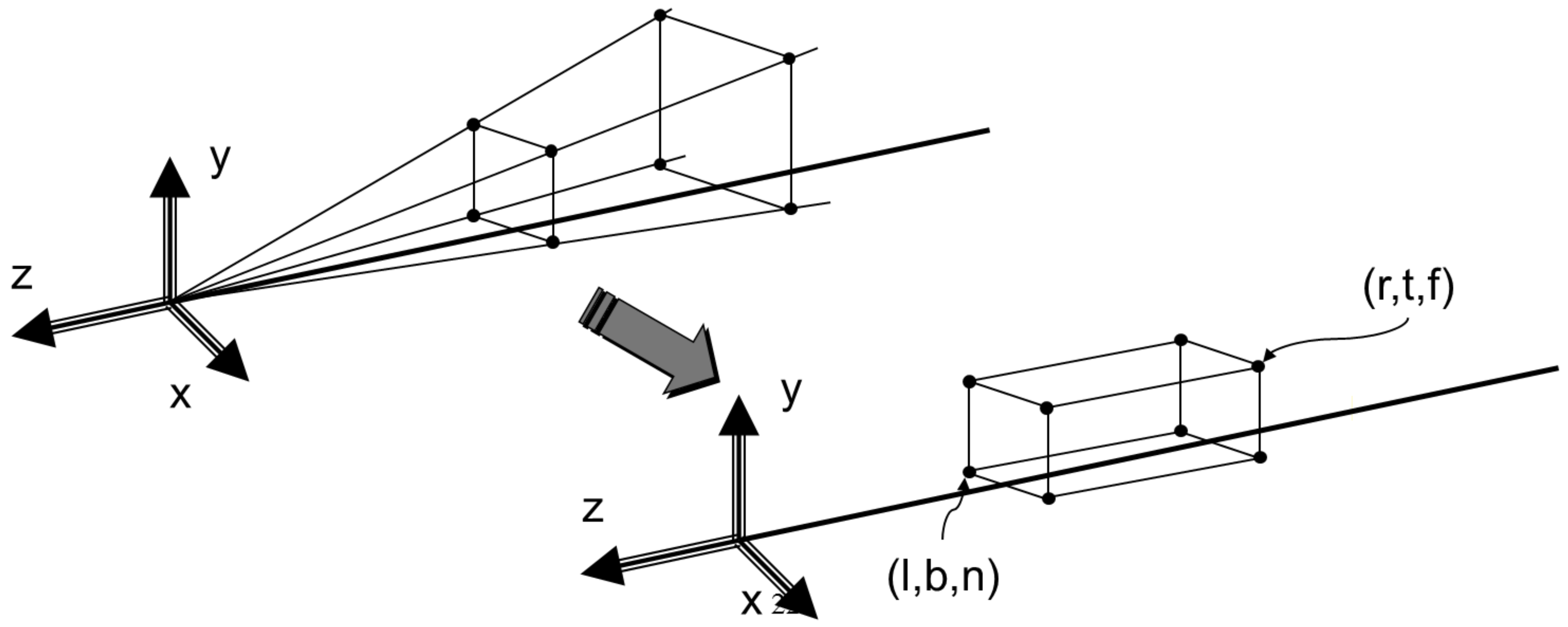
$$y_s = \frac{d}{z} y$$



Projections

Perspective projection

- Leave points on z = n plane (view plane)
- Between z = n and z = f lines trough eye point become parallel to z-axis

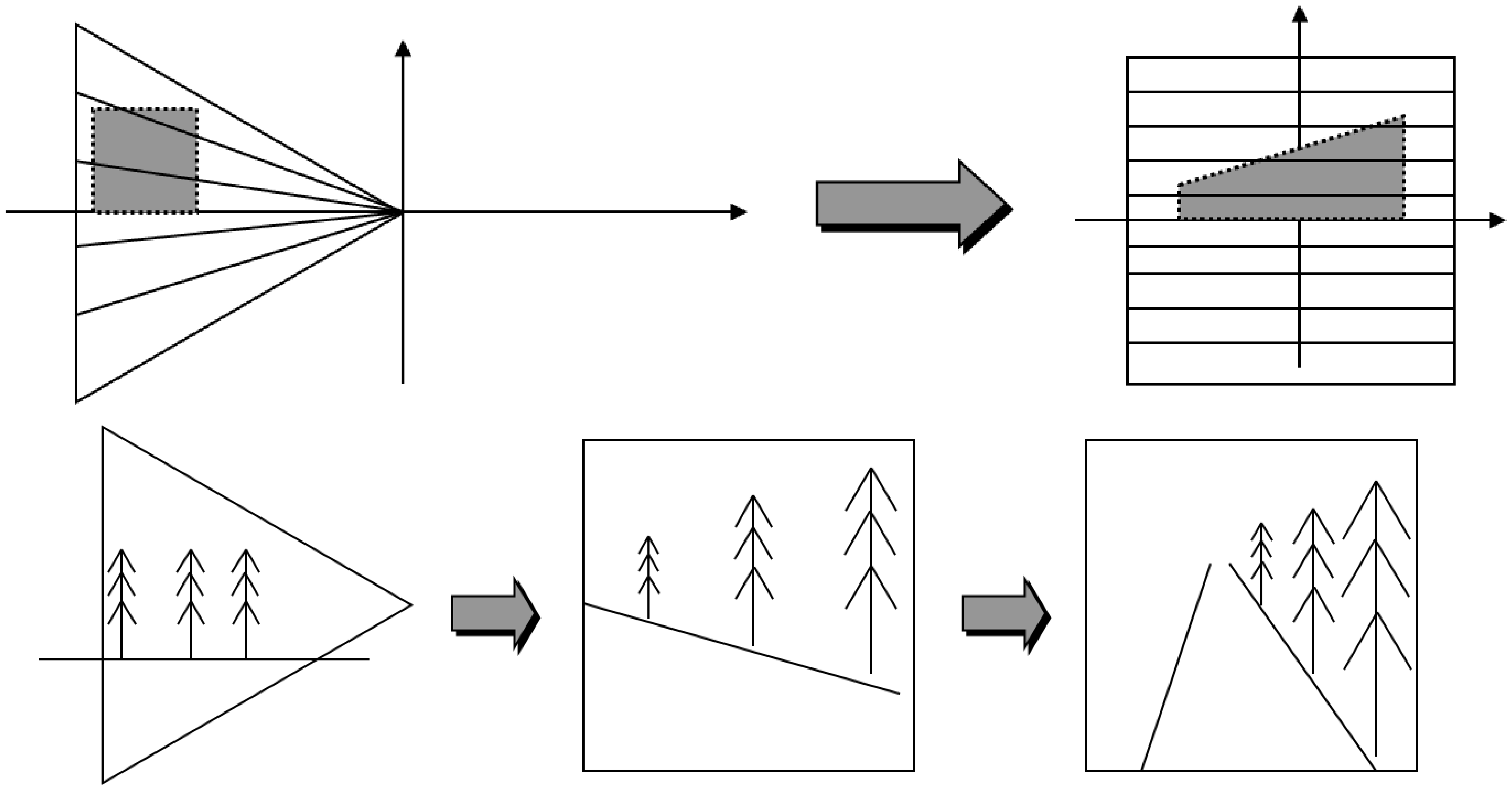


$$M_{perspective} = \begin{pmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

Projections

Normalizing transform

- Regions close to observer enlarged, distant regions shrink
- Perspective distortion



ohm

Projections

Viewport transformation

- Maps the projected 2D coordinates to a screen area in the window