Juraj Onderik | onderik@sccg.sk

Transformations

Projections

Lesson

04

Outline of Lesson 04

- Linear Transformations
- Affine Transformations
- Perspective Projections
- Parallel Projections

Linear Transformations

- Function $L: R^n \rightarrow R^m$ is **linear** iff
  - $L(\mathbf{u} + \mathbf{v}) = L(\mathbf{u}) + L(\mathbf{v})$ (addition)
  - $L(c\mathbf{u}) = cL(\mathbf{u})$ (scalar multiplication)
- Linear function preserves linear combinations
  - $L(c_1\mathbf{u}_1 + \ldots + c_n\mathbf{u}_n) = c_1L(\mathbf{u}_1) + \ldots + c_nL(\mathbf{u}_n)$
- Linear function L is a linear transformation iff
  - Inverse function L^{-1} exists (is invertible)

Linear Transformations

- Linear transformation $L: (x_1, \ldots, x_n) \rightarrow (x'_1, \ldots, x'_n)$
  - $x'_1 = c_{11}x_1 + \ldots + c_{1n}x_n$
  - $\ldots$
  - $x'_n = c_{n1}x_1 + \ldots + c_{nn}x_n$

$$\begin{pmatrix} x'_1 \\ \vdots \\ x'_n \end{pmatrix} = \begin{pmatrix} c_{11} & \cdots & c_{1n} \\ \vdots & \ddots & \vdots \\ c_{n1} & \cdots & c_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

- In matrix form
  - $L(\mathbf{x}): \mathbf{x}' \rightarrow M\,\mathbf{x}$
  - $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{x}' = (x'_1, \ldots, x'_n)$
  - M is $(n \times n)$ transformation matrix $M = (c_{ij})$

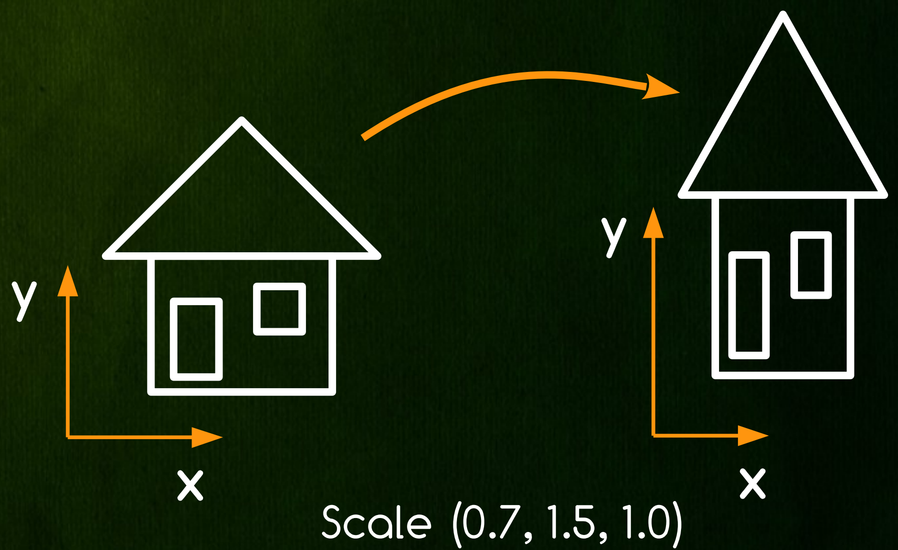
Linear Transformations

- Suppose linear transformations L_1 and L_2
  - $L_1(x) = M_1x$
  - $L_2(x) = M_2x$
- Composite transformation $L(x) = L_2(L_1(x))$
  - $L(x) = L_2(L_1(x)) = L_2(M_1x) = M_2(M_1x) = (M_2M_1)x = Mx$
  - Is linear again: $L(x) = Mx$ where $M = M_2M_1$
  - Is closed under composition $M = M_k..M_1$

Scale

- Scale in 3D by s_x, s_y, s_z
  - $x' = s_x x$
  - $y' = s_y y$
  - $z' = s_z z$
- In Matrix form

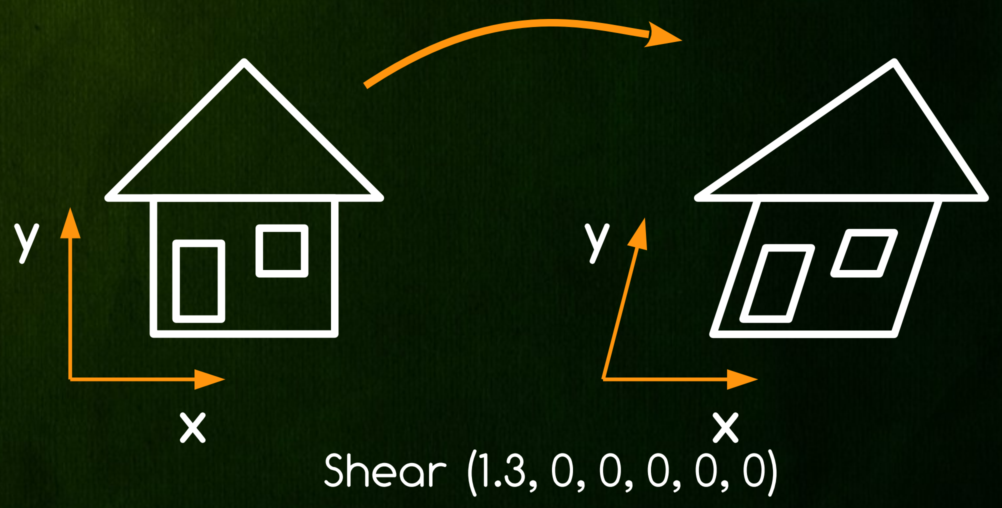
$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_n \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$



Shear

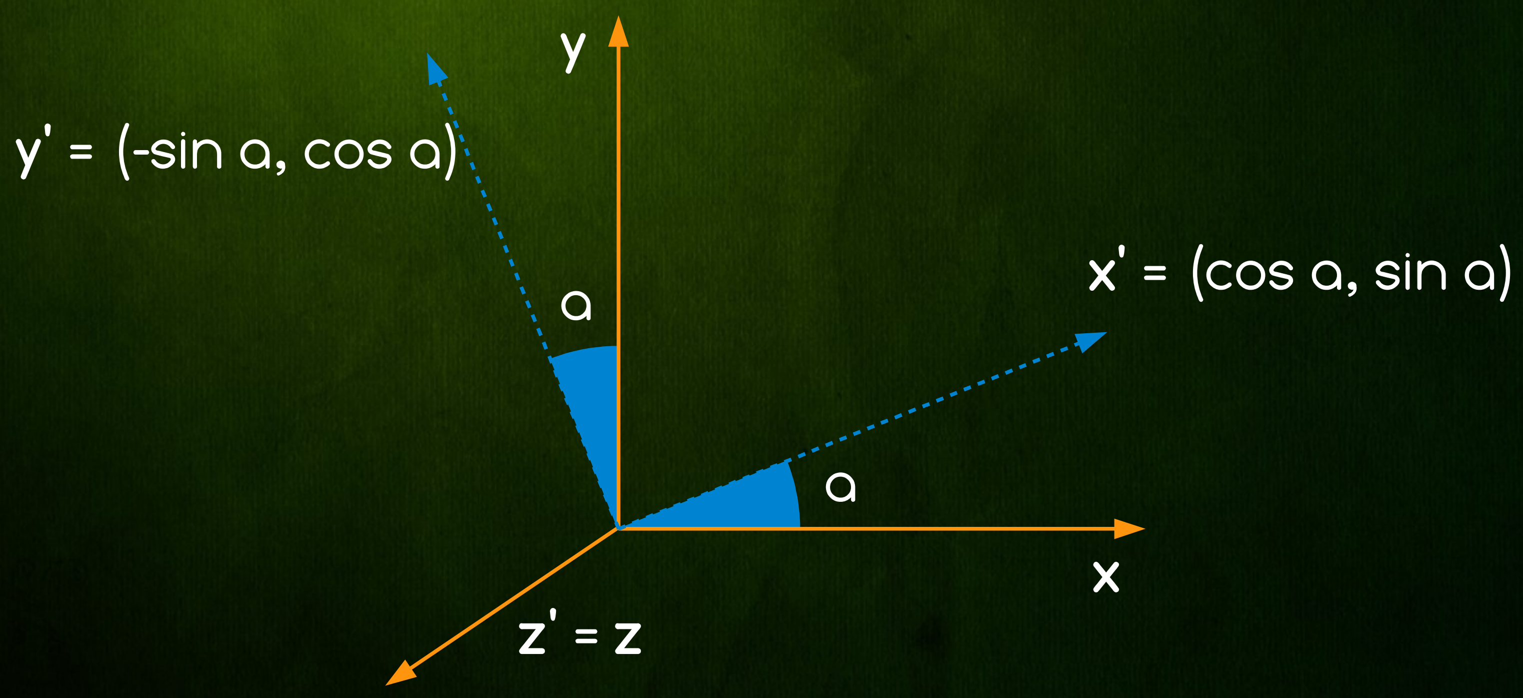
- Shear in 3D by sh_{xy}, sh_{xz}, sh_{yx}, sh_{yz}, sh_{zx} sh_{zy}
  - $x' = x + sh_{xy}y + sh_{xz}z$
  - $y' = s_{yx}x + y + sh_{yz}z$
  - $z' = s_{z}z + sh_{yz}z + z$
- In Matrix form

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} 1 & sh_{xy} & sh_{xz} \\ sh_{yx} & 1 & sh_{yz} \\ sh_{zx} & sh_{zy} & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$



Rotation about Coordinate Axis

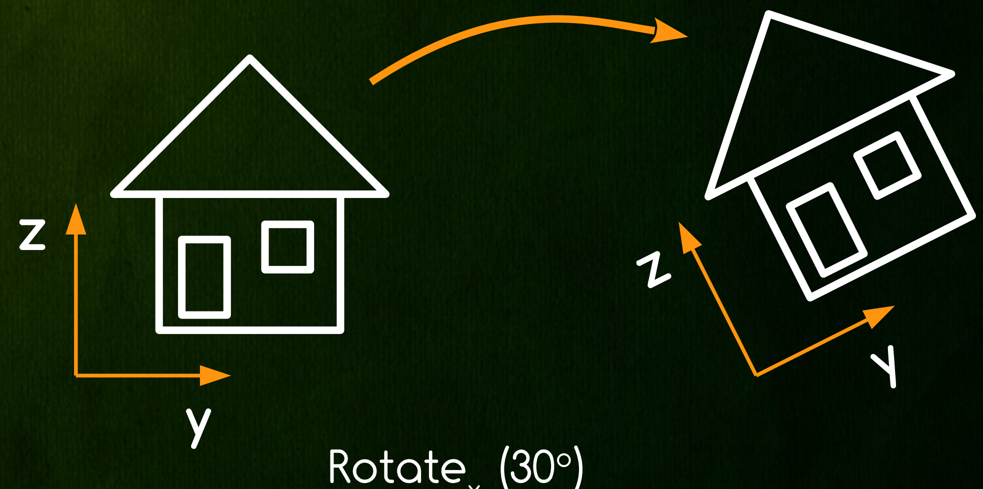
- Rotation about Z-axis



X-Axis Rotation

- Rotation about X-axis in 3D by angle a_x
  - $x' = x$
  - $y' = \cos(a_x)y - \sin(a_x)z$
  - $z' = \sin(a_x)y + \cos(a_x)z$
- In Matrix form

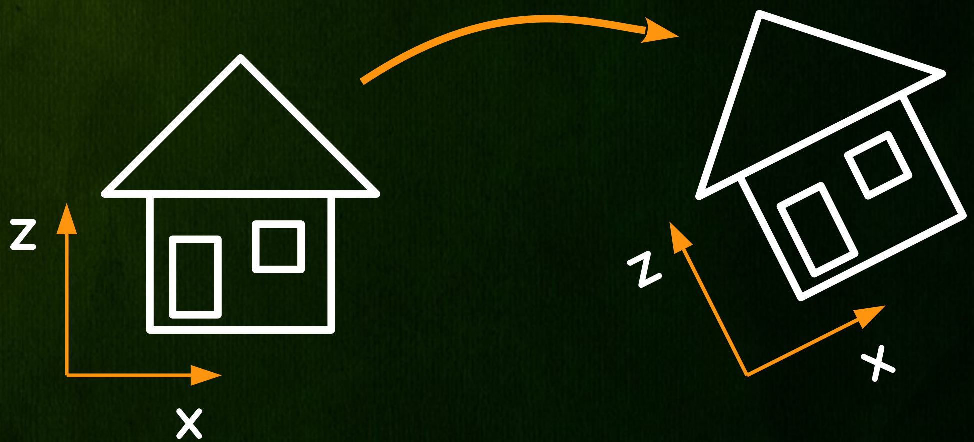
$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & +\cos\alpha & -\sin\alpha \\ 0 & +\sin\alpha & +\cos\alpha \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$



Y-Axis Rotation

- Rotation about Y-axis in 3D by angle a_y
  - $x' = \cos(a_y)x + \sin(a_y)z$
  - $y' = y$
  - $z' = -\sin(a_y)x + \cos(a_y)z$
- In Matrix form

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} +\cos\alpha & 0 & +\sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & +\cos\alpha \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

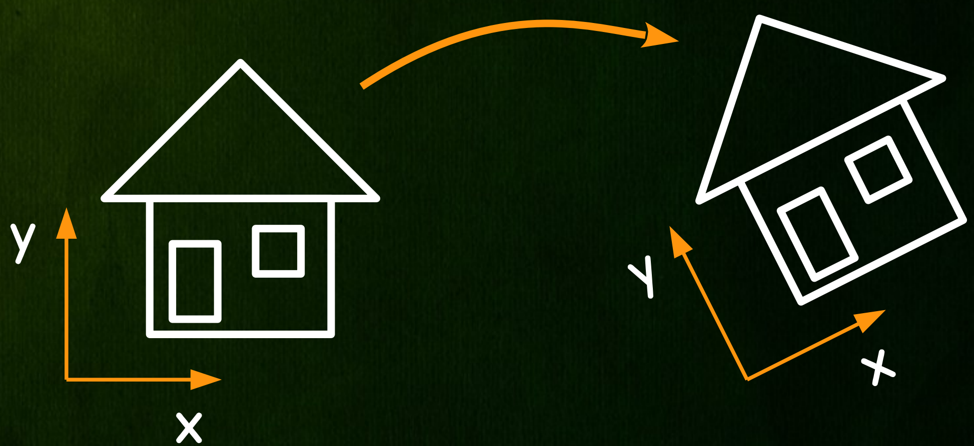


$Rotate_z$ (30°)

Z-Axis Rotation

- Rotation about X-axis in 3D by angle a_x
  - $x' = \cos(a_z)x - \sin(a_z)y$
  - $y' = \sin(a_z)x + \cos(a_z)y$
  - $z' = z$
- In Matrix form

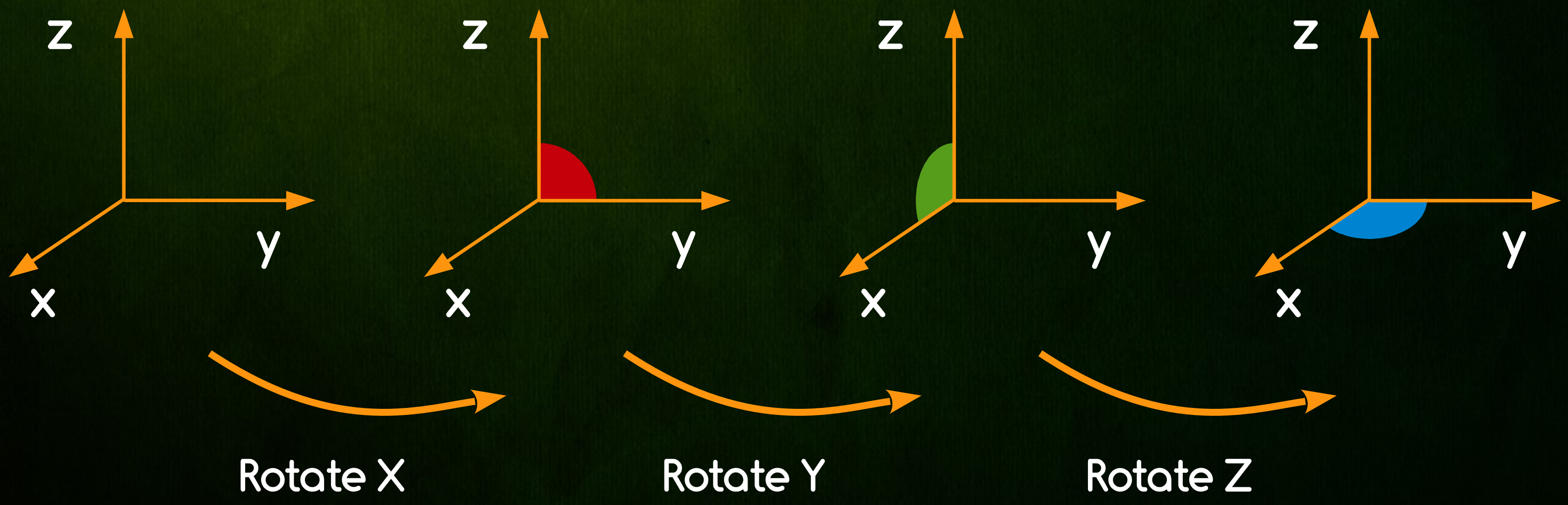
$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} +\cos\alpha & -\sin\alpha & 0 \\ +\sin\alpha & +\cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$



$Rotate_z(30°)$

XYZ Rotation

- XYZ Rotation (a_x, a_y, a_z) is composite rotation around X-axis then by Y-axis and finally Z-axis
  - $R(v) = R_z(R_y(R_x(v))) = R_zR_yR_xv = Rv$
  - $R = R_zR_yR_x$ (matrix multiplication)



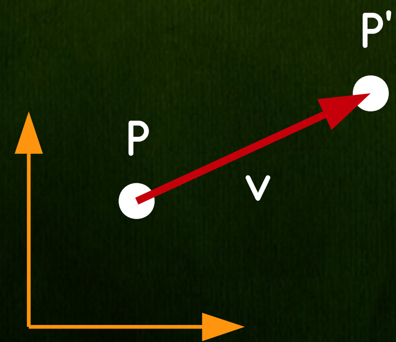
Linear Transformation Summary

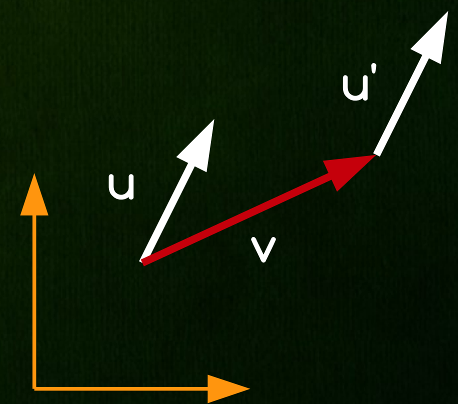
- Origin maps to origin
- Lines map to lines
- Parallel lines remain parallel
- Rotations are preserved
- Closed under composition...

- However simple **translation** can not be defined with linear transformation → we need affine transformations

What is Translation

- What is actually translation ?
- Translation of point **P** by a vector **v** is new point **P'** (= **P** + **v**)
- Translation of vector **u** by a vector **v** is the same vector **v'** (=**v**)





Affine Transformations

- Affine transformation $A: (x_1, \ldots, x_n) \rightarrow (x'_1, \ldots, x'_n)$
  - $x'_1 = c_{11}x_1 + \ldots + c_{1n}x_n + t_1$
  - …
  - $x'_n = c_{n1}x_1 + \ldots + c_{nn}x_n + t_n$

$$\begin{pmatrix} x'_1 \\ \vdots \\ x'_n \end{pmatrix} = \begin{pmatrix} c_{11} & \cdots & c_{1n} \\ \vdots & \ddots & \vdots \\ c_{n1} & \cdots & c_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} + \begin{pmatrix} t_1 \\ \vdots \\ t_n \end{pmatrix}$$

- In a "translation" form
  - $A(\mathbf{x}): \mathbf{x}' \rightarrow \mathbf{M}\,\mathbf{x} + \mathbf{t}$ (= linear transform. + translation)
  - $\mathbf{x}' = (x'_1, \ldots, x'_n)$ | $\mathbf{x} = (x_1, \ldots, x_n)$ | $t = (t'_1, \ldots, t'_n)$
  - M is $(n \times n)$ transformation matrix $M = (c_{ij})$

Affine Transformations

- Can we find pure matrix form ?
- Yes, we need homogenous coordinates
  - Use one more dimension ($\mathbf{R}^{n+1}$)
  - Points: $\mathbf{p} = (p_1, \ldots, p_n)$ become $(p_1, \ldots, p_n, 1)$
  - Vectors: $\mathbf{v} = (v_1, \ldots, v_n)$ become $(v_1, \ldots, v_n, 0)$
- Matrix form

$$\begin{pmatrix} p'_1 \\ \vdots \\ p'_n \\ 1 \end{pmatrix} = \begin{pmatrix} c_{11} & \cdots & c_{1n} & t_1 \\ \vdots & \ddots & \vdots & \vdots \\ c_{n1} & \cdots & c_{nn} & t_n \\ 0 & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} p_1 \\ \vdots \\ p_n \\ 1 \end{pmatrix} \qquad \begin{pmatrix} v'_1 \\ \vdots \\ v'_n \\ 0 \end{pmatrix} = \begin{pmatrix} c_{11} & \cdots & c_{1n} & t_1 \\ \vdots & \ddots & \vdots & \vdots \\ c_{n1} & \cdots & c_{nn} & t_n \\ 0 & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} v_1 \\ \vdots \\ v_n \\ 0 \end{pmatrix}$$

Translation in Matrix form

- Translation of point (or vector) $x' = x + t$
  - $x' = (x'_1, \ldots, x'_n, x'_{n+1})$, $x = (x_1, \ldots, x_n, x_{n+1})$, $t = (t_1, \ldots, t_n, 0)$
  - $x_1 = x_1 + t_1$ | … | $x_n = x_n + t_n$
- Can be expressed in matrix form as
  - $x' = T\,x$
  - T – is translation matrix ($R^{n+1} \times R^{n+1}$)

$$\begin{pmatrix} x'_1 \\ \vdots \\ x'_n \\ x'_{n+1} \end{pmatrix} = \begin{pmatrix} 1 & \cdots & 0 & t_1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & t_n \\ 0 & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \\ x_{n+1} \end{pmatrix}$$

$$x' \quad = \quad T \quad x$$

Affine Transformations

- Using homogenous coordinates we can
  - Express linear transformation **M** and translation **T**

$$\mathbf{M}=\begin{pmatrix} c_{11} & \cdots & c_{1\mathrm{n}} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ c_{n1} & \cdots & c_{nn} & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}, \qquad \mathbf{T}=\begin{pmatrix} 1 & \cdots & 0 & t_1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & t_n \\ 0 & \cdots & 0 & 1 \end{pmatrix}$$

- Therefore A(x) = Mx + t = T(Mx) = TMx

$$\begin{pmatrix} x'_1 \\ \vdots \\ x'_n \\ x'_{n+1} \end{pmatrix}=\begin{pmatrix} 1 & \cdots & 0 & t_1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & t_n \\ 0 & \cdots & 0 & 1 \end{pmatrix}\begin{pmatrix} c_{11} & \cdots & c_{1\mathrm{n}} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ c_{n1} & \cdots & c_{nn} & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}\begin{pmatrix} x_1 \\ \vdots \\ x_n \\ x_{n+1} \end{pmatrix}=\begin{pmatrix} c_{11} & \cdots & c_{1\mathrm{n}} & t_1 \\ \vdots & \ddots & \vdots & \vdots \\ c_{n1} & \cdots & c_{nn} & t_n \\ 0 & \cdots & 0 & 1 \end{pmatrix}\begin{pmatrix} x_1 \\ \vdots \\ x_n \\ x_{n+1} \end{pmatrix}$$

Basic Transformations

- **Rotation around major axis**

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

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

  - Assumes a right handed coordinate system

Basic Transformations

- **Scaling**

$$S(s_x, s_y, s_z) = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

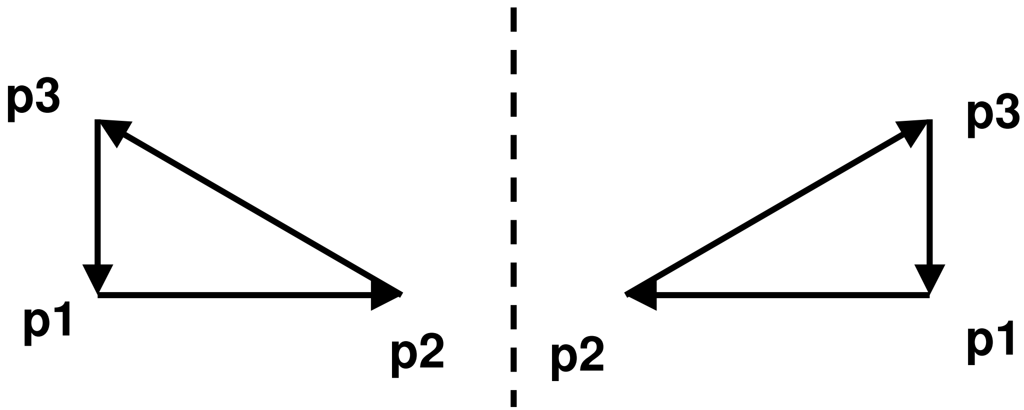
  – Uniform Scaling
    - $s_x = s_y = s_z$

Basic Transformations

- **Reflection at Z**

$$M_z = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

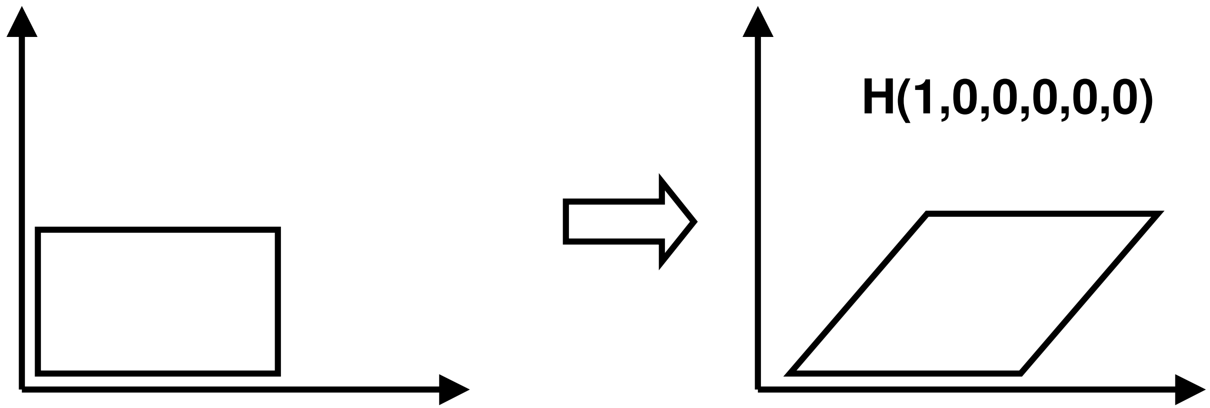
  - Warning: Change of orientation !



Basic Transformations

- **Shear (deutsch: Scherung)**

$$H(h_{xy}, h_{xz}, h_{yz}, h_{yx}, h_{zx}, h_{zy}) = \begin{pmatrix} 1 & h_{xy} & h_{xz} & 0 \\ h_{yx} & 1 & h_{yz} & 0 \\ h_{zx} & h_{zy} & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

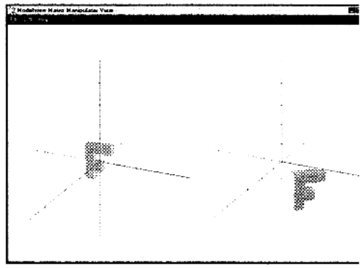


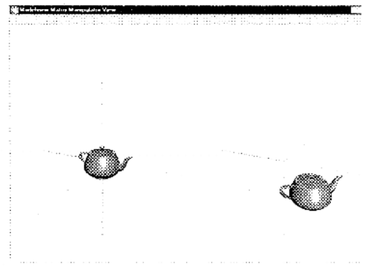
Concatenation of Transformations

- **Matrix multiplication**
  - Read transformations from right to left !

(e) Rotation followed by translation

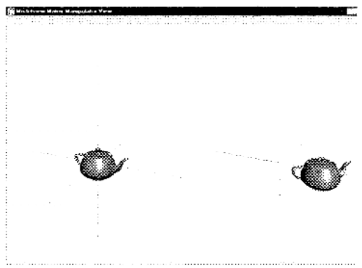
$$\begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.866 & 0.5 & 0 & 0 \\ -0.5 & 0.866 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.866 & 0.5 & 0 & 2 \\ -0.5 & 0.866 & 0 & 2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

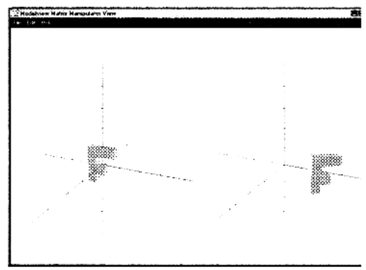




(f) Translation followed by rotation

$$\begin{bmatrix} 0.866 & 0.5 & 0 & 0 \\ -0.5 & 0.866 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.866 & 0.5 & 0 & 2.732 \\ -0.5 & 0.866 & 0 & 0.732 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$





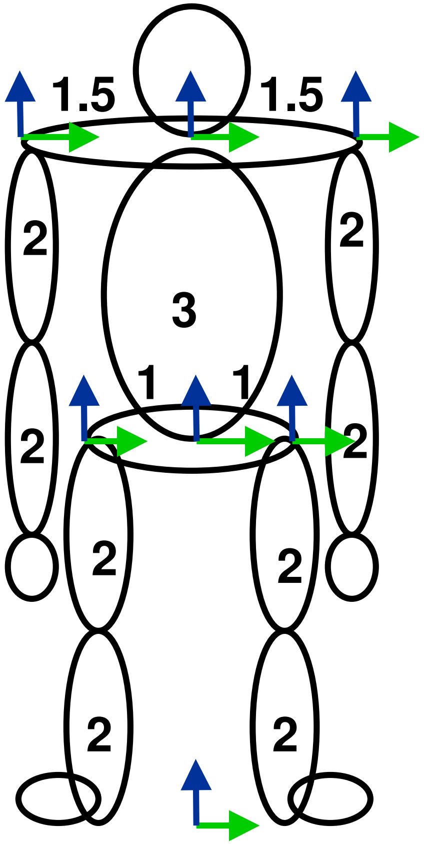
Affine Transformation Summary

- Origin **does not** map to origin
- Lines map to lines
- Parallel lines remain parallel
- Rotations are preserved
- Closed under composition...
- Translation can be expressed

Coordinate Systems

- **Object Coordinates**
  - Intrinsic coordinate system of an object
  - Hierarchical modeling
  - Modeling Transformations to world coordinates
- **World Coordinates**
  - Root for hierarchical modeling
  - Reference system for the camera
  - Viewing Transformation to camera coordinates
- **Camera coordinates (Viewing Coordinates)**
  - Reference system for lighting computations
  - Perspective Transformation to normalized (projection) coordinates

Hierarchical Modeling




```
body
  torso
    head
    shoulder
      larm
        upperarm
        lowerarm
        hand
      rarm
        upperarm
        lowerarm
        hand
    hips
      lleg
        upperleg
        lowerleg
        foot
      rleg
        upperleg
        lowerleg
        foot
```

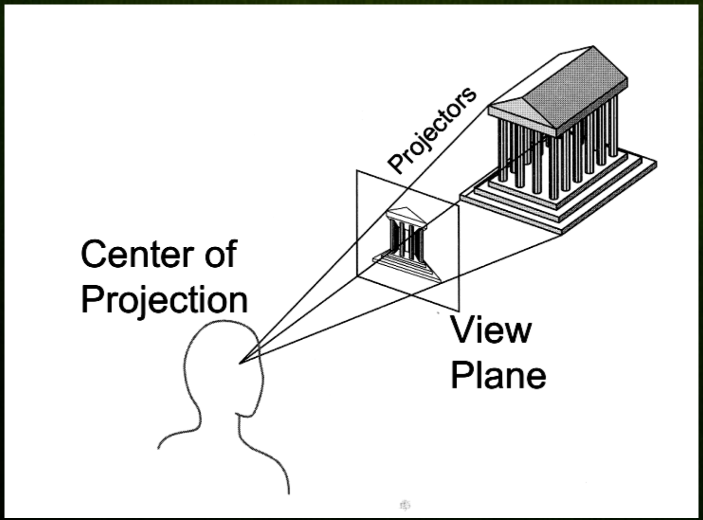
```
Translate 0 4 0
TransformBegin
  # Draw Torso
  Translate 0 3 0
  # Draw Shoulders
  TransformBegin
    Rotate a 0 0 1
    # Draw head
  TransformEnd
  TransformBegin
    Translate 1.5 0 0
    DRAW_ARM(a,b,c)
TransformEnd
TransformBegin
    Translate -1.5 0 0
    DRAW_ARM(d,e,f)
  TransformEnd
TransformEnd
# Draw hips
TransformBegin
  TransformBegin
    Translate 1 0 0
    DRAW_LEG(g,h)
TransformEnd
  TransformBegin
    Translate -1 0 0
    DRAW_LEG(i,j)
  TransformEnd
TransformEnd
```

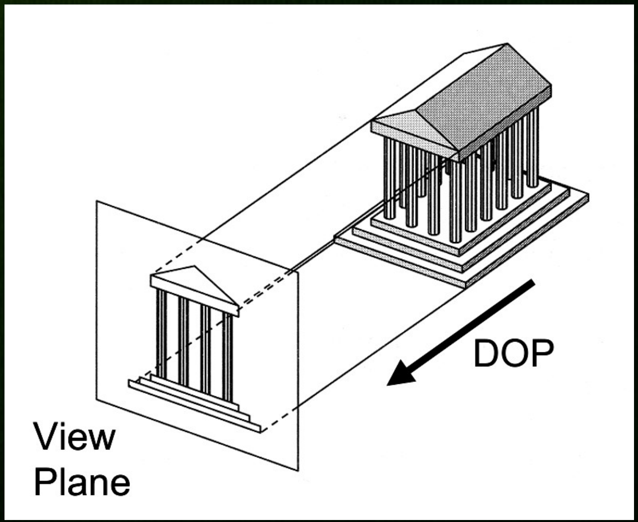
```
DRAW_ARM(a,b,c) {
      Rotate b 0 0 1
      # Draw upperarm
      Translate 0 -2 0
      Rotate c 1 0 0
      # Draw lowerarm
      Translate 0 -2 0
      # Draw hand
     }

DRAW_LEG(g,h) {
    Rotate g 1 0 0
    # Draw upperleg
    Translate 0 -2 0
    Rotate h 1 0 0
    # Draw lowerleg
    Translate 0 -2 0
    # Draw foot
    }
```

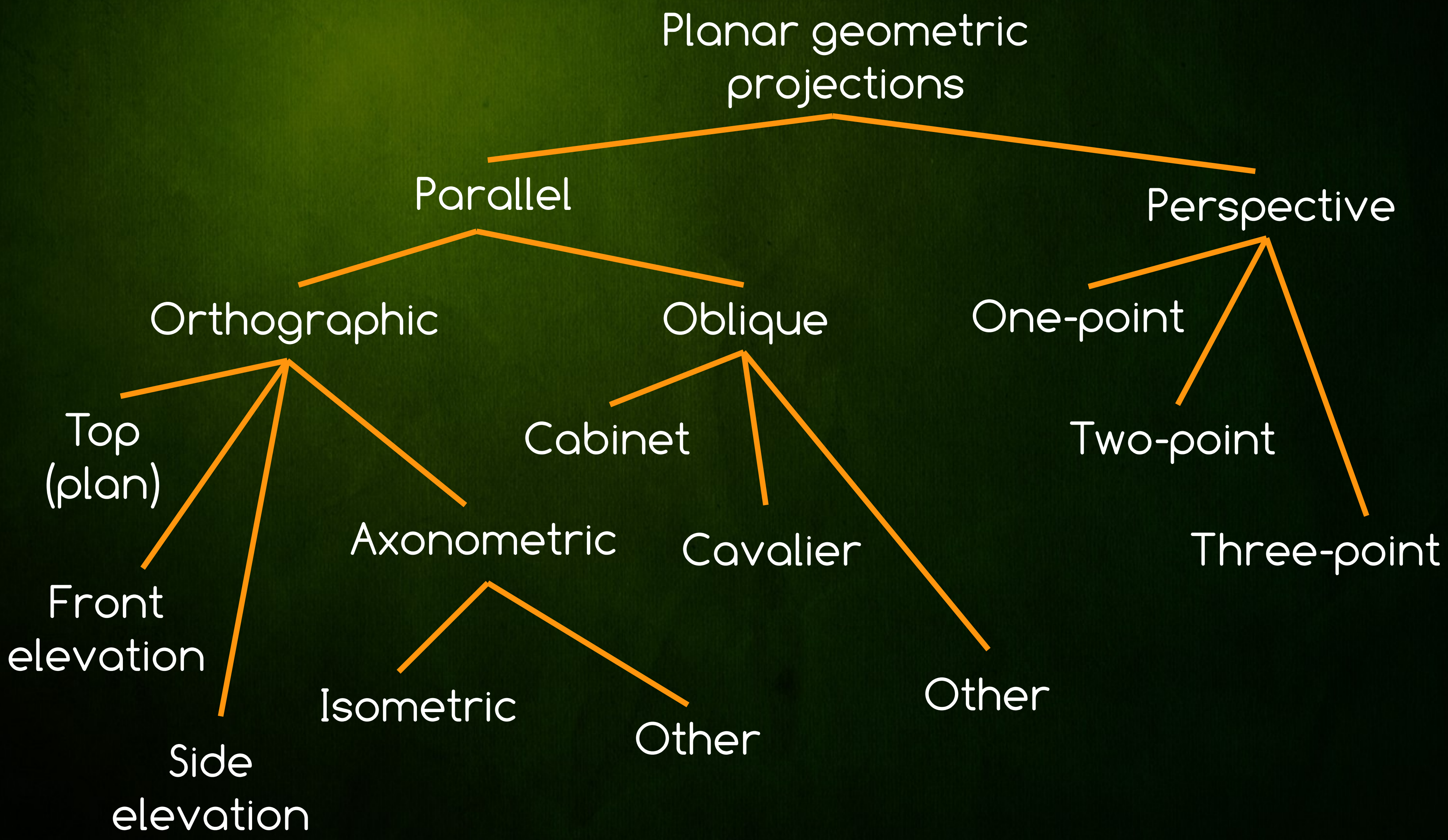
Projections

- General definition
  - Transform points in n-space to m-space (m<n)
- In computer graphics
  - Map 3D camera coordinates to 2D screen coordinates

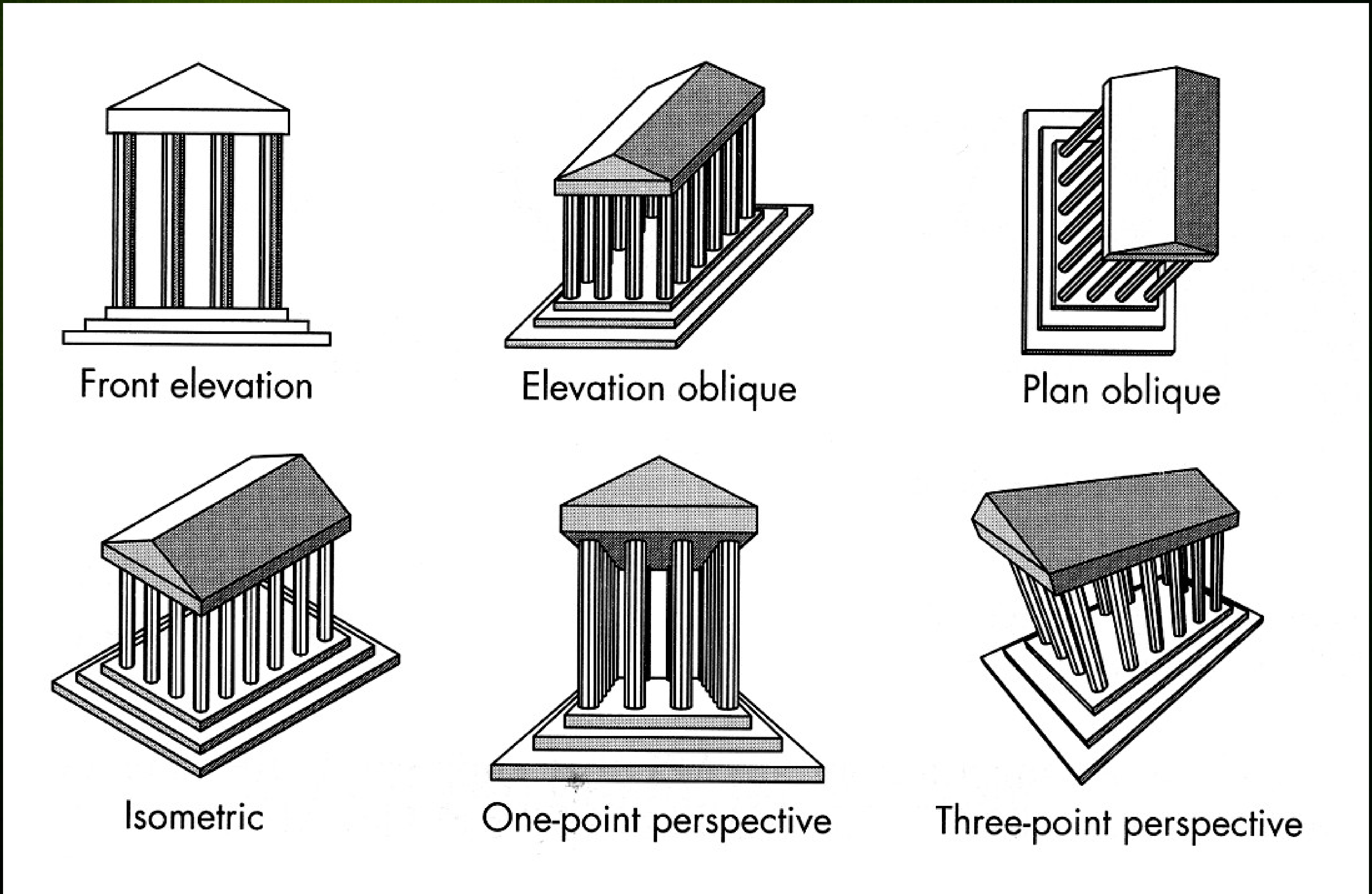




Taxonomy Projections

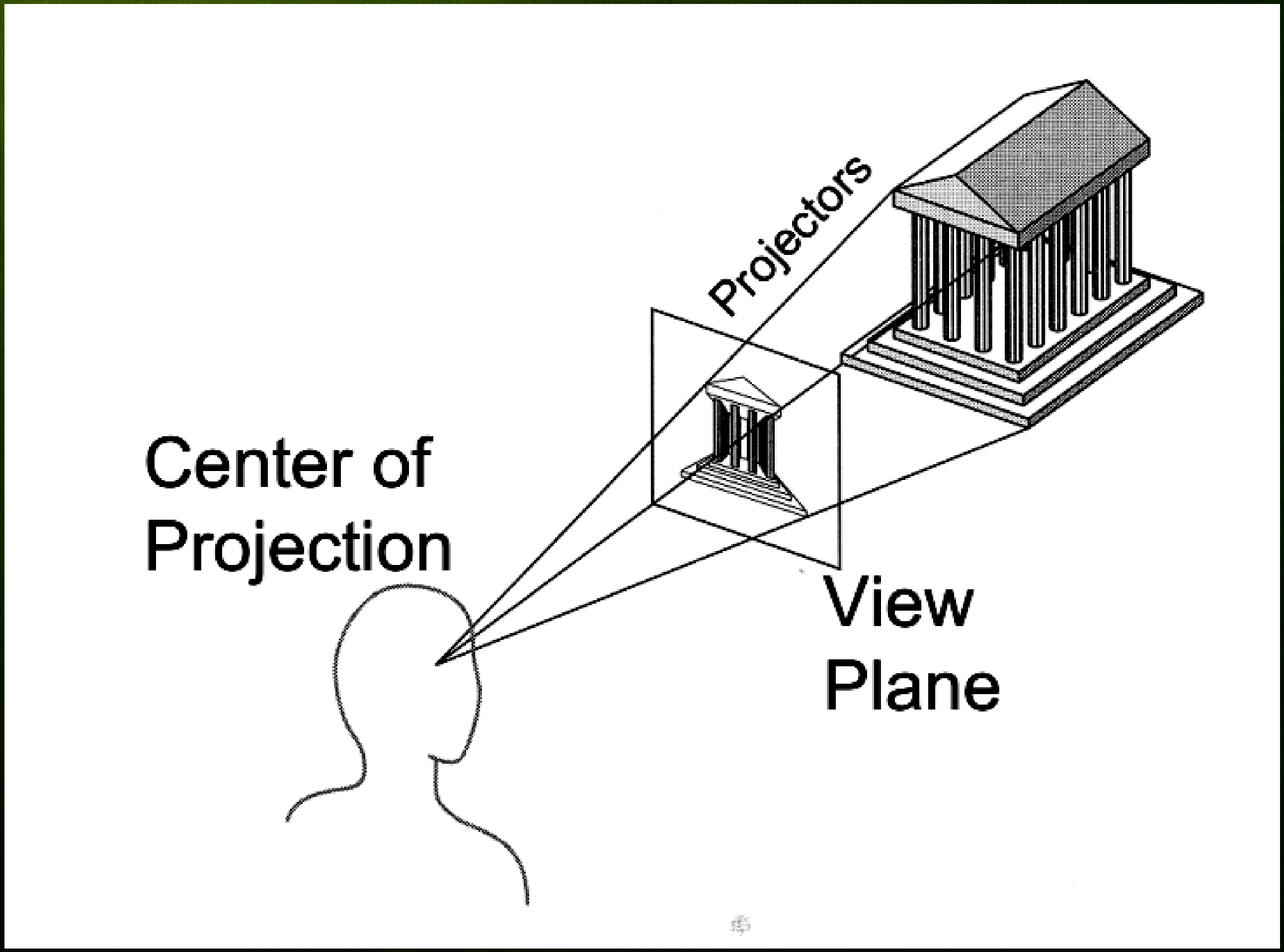


Projection Types



Perspective Projection

- Map points onto “view plane” along “projectors” emanating from “center of projection” (COP)

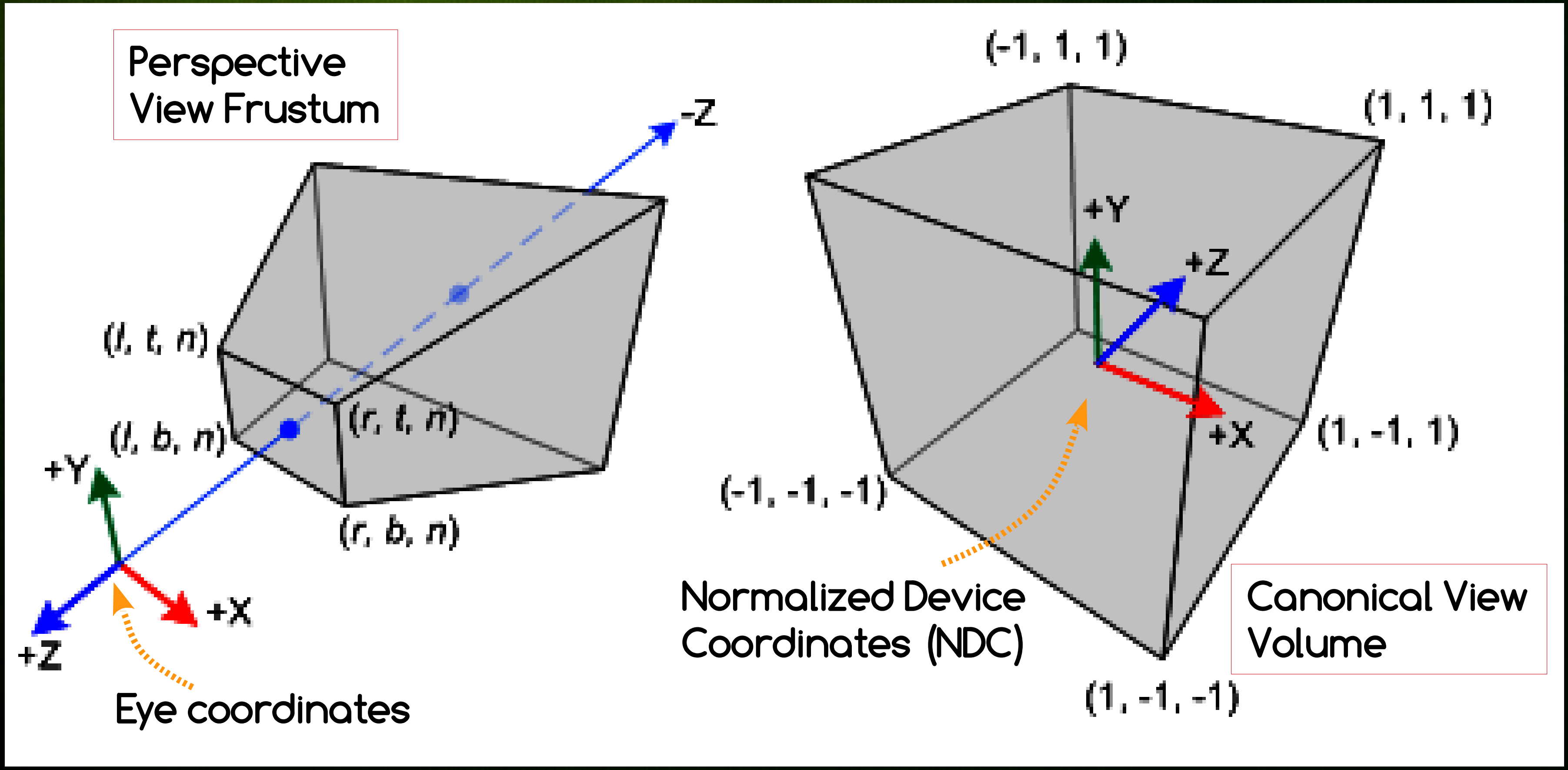


Perspective Projection

- In perspective projection, a 3D point in
- a truncated pyramid - view frustum (in eye coordinates) is mapped to
- a cube (Normalized device coordinates)
  - The x-coordinate from [l, r] to [-1, 1]
  - The y-coordinate from [b, t] to [-1, 1]
  - The z-coordinate from [n, f] to [-1, 1].

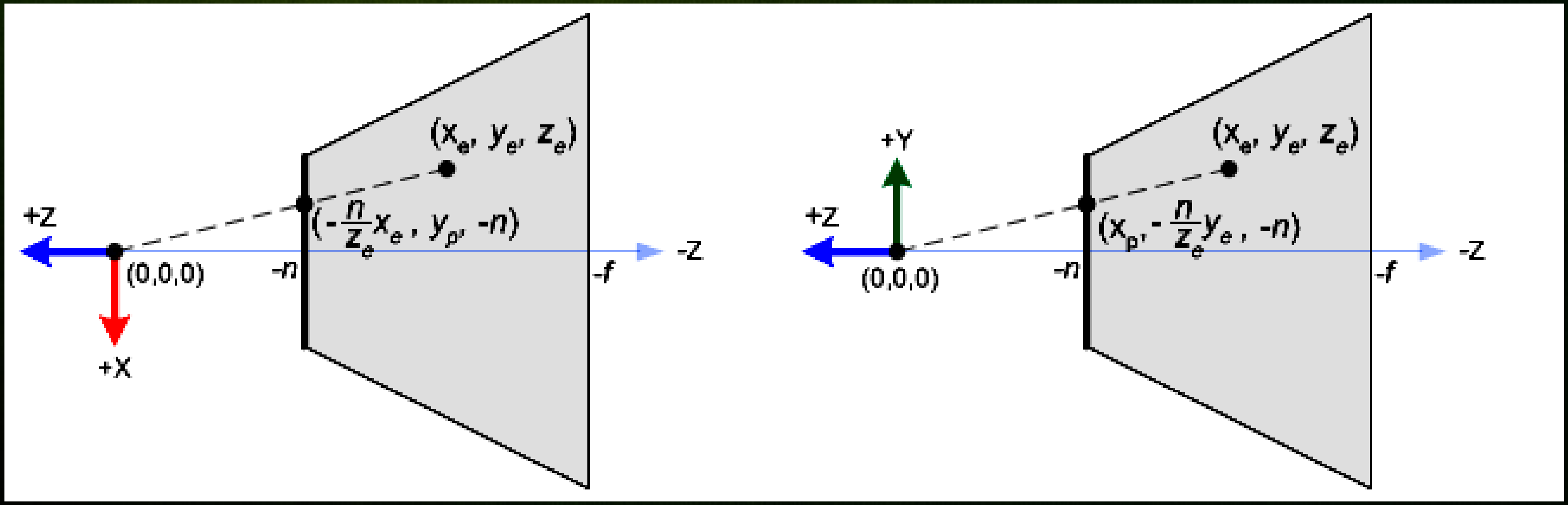
Perspective View Frustum

- Definition of perspective view frustum
  - l (left), r (right), b (bottom), t (top), n (near), f (far)



Perspective Projection

- Eye to near plane projection $(x_e, y_e, z_e) \rightarrow (x_p, y_p, z_p)$
  - Similar triangles ratio $x_p/x_e = -n/z_e \rightarrow x_p = -(n/z_e)x_e$
  - Similar triangles ratio: $y_p/y_e = -n/z_e \rightarrow y_p = -(n/z_e)y_e$
  - We project on near plane $\rightarrow z_p = -n$



Perspective Projection

- Since projected point (x_p, y_p, z_p) has division in its definition there is no matrix formulation

- We split Perspective Projection into
  - 1) Homogenous perspective projection P
  - 2) Clip projection C

Perspective Projection Steps

- Homogenous perspective projection
  - From eye coordinates (x_e, y_e, z_e, w_e)
  - To clip coordinates (x_c, y_c, z_c, w_c)
  - 4x4 homogenous transformation matrix P

$$\begin{pmatrix} x_c \\ y_c \\ z_c \\ w_c \end{pmatrix} = \begin{pmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{pmatrix} \begin{pmatrix} x_e \\ y_e \\ z_e \\ w_e \end{pmatrix}$$

Perspective Projection Steps

- Clip projection
  - From homogenous clip coordinates (x_e, y_e, z_e, w_e)
  - To normalized device coordinates (x_n, y_n, z_n)
  - Reduction from homogenous coordinates to normal 3d coordinates

$$\begin{pmatrix} x_n \\ y_n \\ z_n \end{pmatrix} = \begin{pmatrix} x_c / w_c \\ y_c / w_c \\ z_c / w_c \end{pmatrix}$$

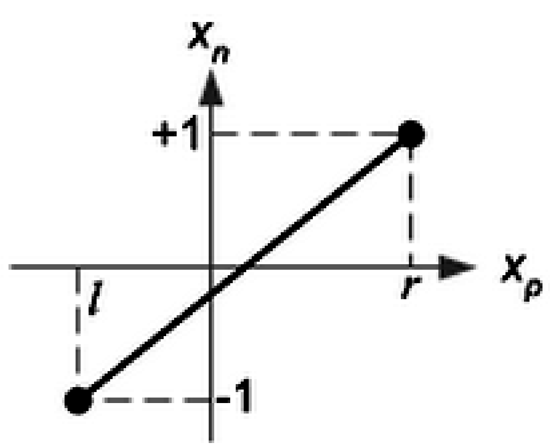
Perspective Projection

- Since x_p and y_p are inverse proportional to $-z_e$
- We set $w_c = -z_e$ to postpone division by $-z_e$ into Clip projection
- Therefore last row of homogenous projection matrix P is (0,0,-1,0)

$$\begin{pmatrix} x_c \\ y_c \\ z_c \\ w_c \end{pmatrix} = \begin{pmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} x_e \\ y_e \\ z_e \\ w_e \end{pmatrix}$$

Perspective Projection

- Map x_p and y_p to x_n and y_n of NDC with linear interpolation [l, r] → [-1, 1] and [b, t] → [-1, 1]



Mapping from x_p to x_n

$$x_n = \frac{1-(-1)}{r-l} \cdot x_p + \beta$$

$$1 = \frac{2r}{r-l} + \beta \qquad (\text{substitute } (r, 1) \text{ for } (x_p, x_n))$$

$$\beta = 1 - \frac{2r}{r-l} = \frac{r-l}{r-l} - \frac{2r}{r-l}$$

$$= \frac{r-l-2r}{r-l} = \frac{-r-l}{r-l} = -\frac{r+l}{r-l}$$

$$\therefore x_n = \frac{2x_p}{r-l} - \frac{r+l}{r-l}$$

$$\therefore y_n = \frac{2y_p}{t-b} - \frac{t+b}{t-b}$$

Perspective Projection

$$\begin{aligned}
x_n &= \frac{2x_p}{r-l} - \frac{r+l}{r-l} \qquad \left(x_p = \frac{nx_e}{-z_e}\right) \\
&= \frac{2 \cdot \frac{n \cdot x_e}{-z_e}}{r-l} - \frac{r+l}{r-l} \\
&= \frac{2n \cdot x_e}{(r-l)(-z_e)} - \frac{r+l}{r-l} \\
&= \frac{\frac{2n}{r-l} \cdot x_e}{-z_e} - \frac{r+l}{r-l} \\
&= \frac{\frac{2n}{r-l} \cdot x_e}{-z_e} + \frac{\frac{r+l}{r-l} \cdot z_e}{-z_e} \\
&= \underbrace{\left(\frac{2n}{r-l} \cdot x_e + \frac{r+l}{r-l} \cdot z_e \right)}_{x_c} \Big/ -z_e
\end{aligned}$$

$$\begin{aligned}
y_n &= \frac{2y_p}{t-b} - \frac{t+b}{t-b} \qquad \left(y_p = \frac{ny_e}{-z_e}\right) \\
&= \frac{2 \cdot \frac{n \cdot y_e}{-z_e}}{t-b} - \frac{t+b}{t-b} \\
&= \frac{2n \cdot y_e}{(t-b)(-z_e)} - \frac{t+b}{t-b} \\
&= \frac{\frac{2n}{t-b} \cdot y_e}{-z_e} - \frac{t+b}{t-b} \\
&= \frac{\frac{2n}{t-b} \cdot y_e}{-z_e} + \frac{\frac{t+b}{t-b} \cdot z_e}{-z_e} \\
&= \underbrace{\left(\frac{2n}{t-b} \cdot y_e + \frac{t+b}{t-b} \cdot z_e \right)}_{y_c} \Big/ -z_e
\end{aligned}$$

Perspective Projection

- z_n and z_c do not depend on x_e and y_e thus

$$\begin{pmatrix} x_c \\ y_c \\ z_c \\ w_c \end{pmatrix} = \begin{pmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & A & B \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} x_e \\ y_e \\ z_e \\ w_e \end{pmatrix}$$

$$z_n = \frac{z_c}{w_c} = \frac{Az_e + Bw_e}{-z_e}$$

- Solve A and B for boundary values of z_e and z_n
  - When $z_e = -n \rightarrow z_n = -1$ | $-An + B = -n$
  - When $z_e = -f \rightarrow z_n = +1$ | $-Af + B = f$
  - Solve A and B from the these 2 linear equations

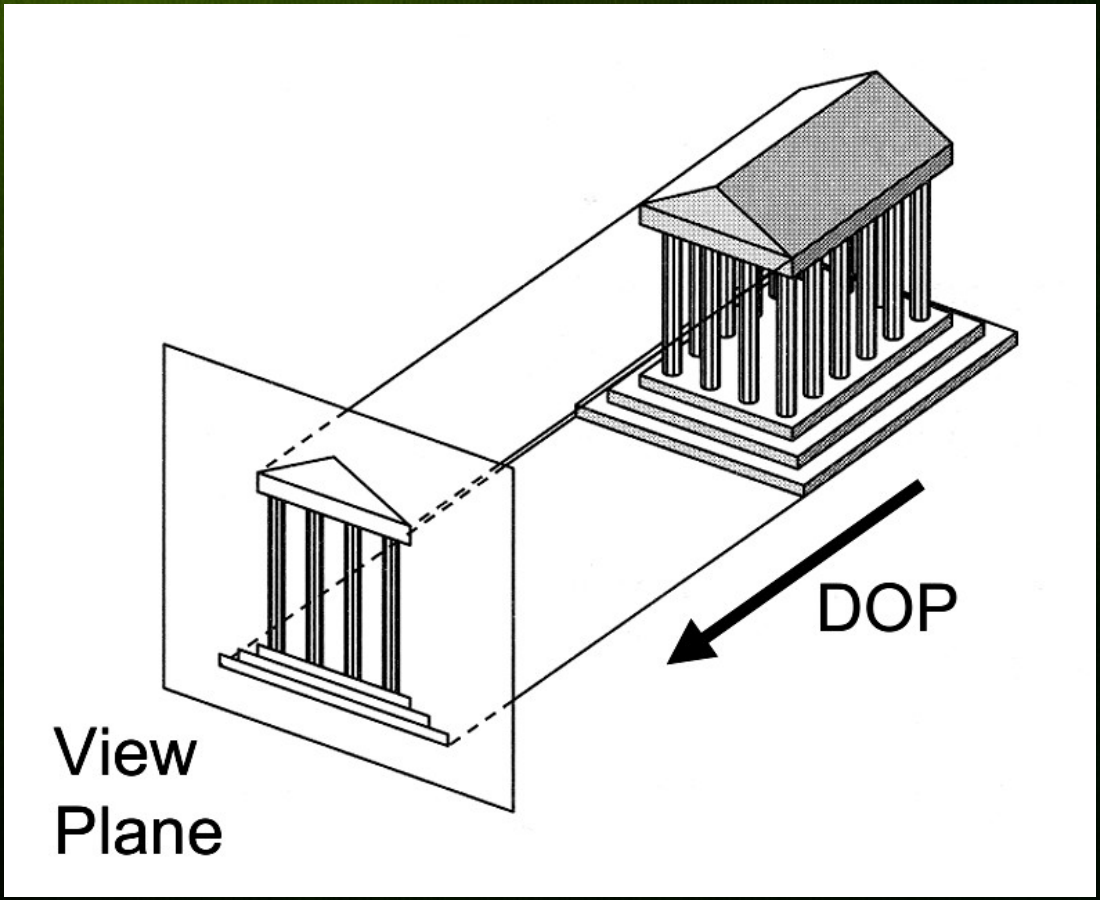
Perspective Projection

- After solving A and B we get
  - A = -(f + n) / (f - n) | B = -2fn / (f – n)
- And we get final Projection Matrix

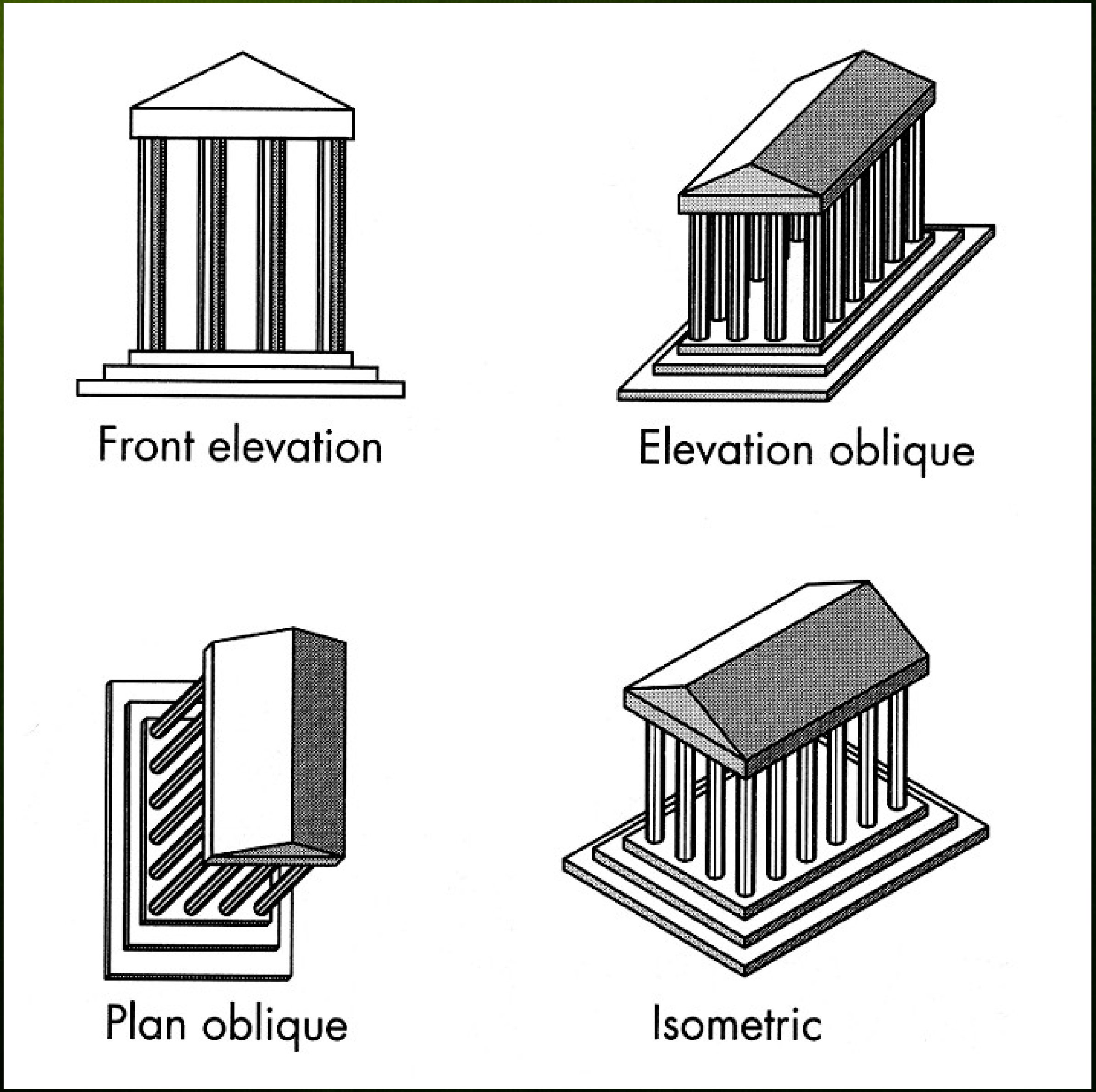
$$\begin{pmatrix} x_c \\ y_c \\ z_c \\ w_c \end{pmatrix} = \begin{pmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & \frac{-(f+n)}{f-n} & \frac{-2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} x_e \\ y_e \\ z_e \\ w_e \end{pmatrix}$$

Parallel Projection

- Center of projection is at infinity ✌
  - Direction of projection (DOP) same for all points

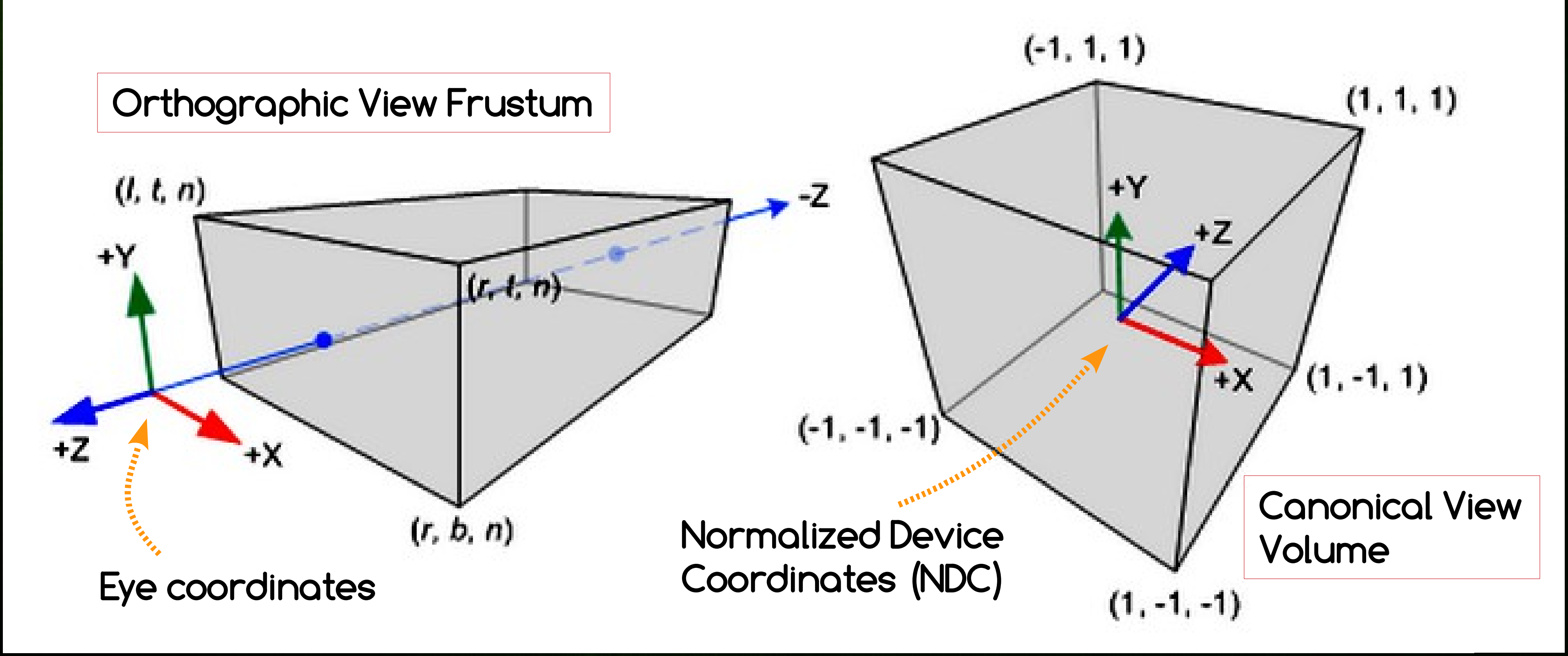


Parallel Projection Types



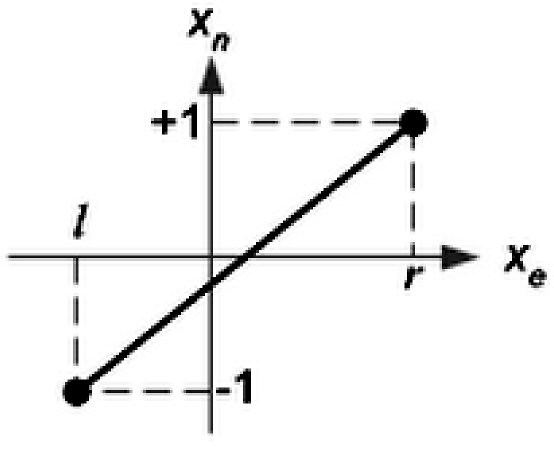
Orthographic Projection

- Definition of orthographic view frustum
  - l (left), r (right), b (bottom), t (top), n (near), f (far)



Orthographic Projection

- No homogenous projection needed
- We transform x_e to x_n with linear interpolation
- We map input interval (l, r) → (-1, +1)



Mapping from x_e to x_n

$$x_n = \frac{1-(-1)}{r-l} \cdot x_e + \beta$$

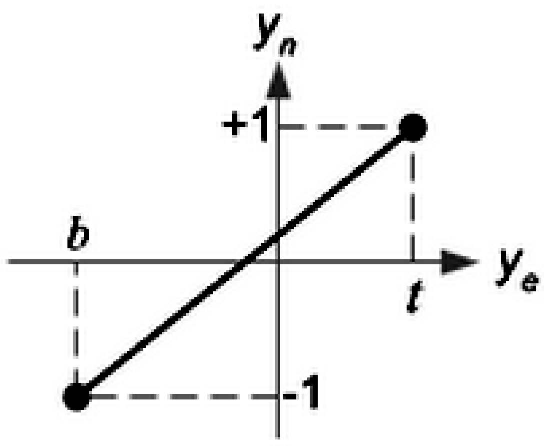
$$1 = \frac{2r}{r-l} + \beta \qquad \text{(substitute } (r, 1) \text{ for } (x_e, x_n))$$

$$\beta = 1 - \frac{2r}{r-l} = -\frac{r+l}{r-l}$$

$$\therefore x_n = \frac{2}{r-l} \cdot x_e - \frac{r+l}{r-l}$$

Orthographic Projection

- No homogenous projection needed
- We transform y_e to y_n with linear interpolation
- We map input interval (b, t) → (-1, +1)



Mapping from y_e to y_n

$$y_n = \frac{1-(-1)}{t-b} \cdot y_e + \beta$$

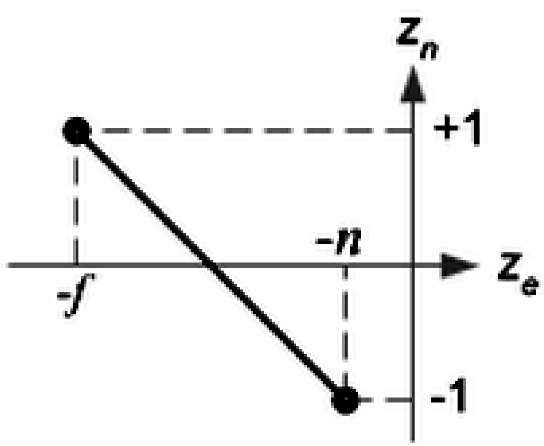
$$1 = \frac{2t}{t-b} + \beta \qquad \text{(substitute } (t, 1) \text{ for } (y_e, y_n))$$

$$\beta = 1 - \frac{2t}{t-b} = -\frac{t+b}{t-b}$$

$$\therefore y_n = \frac{2}{t-b} \cdot y_e - \frac{t+b}{t-b}$$

Orthographic Projection

- No homogenous projection needed
- We transform z_e to z_n with linear interpolation
- We map input interval (-f, -n) → (+1, -1)



Mapping from z_e to z_n

$$z_n = \frac{1-(-1)}{-f-(-n)} \cdot z_e + \beta$$

$$1 = \frac{2f}{f-n} + \beta \qquad \text{(substitute } (-f, 1) \text{ for } (z_e, z_n))$$

$$\beta = 1 - \frac{2f}{f-n} = -\frac{f+n}{f-n}$$

$$\therefore z_n = \frac{-2}{t-b} \cdot z_e - \frac{f+n}{f-n}$$

Orthographic Projection

- Final 4x4 orthographic projection is
- It is affine transformation $w_c = w_e$

$$\begin{pmatrix} x_c \\ y_c \\ z_c \\ w_c \end{pmatrix} = \begin{pmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & \frac{-2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_e \\ y_e \\ z_e \\ w_e \end{pmatrix}$$

Perspective vs. Parallel Projection

- Perspective projection
  - + Size varies inversely with distance - looks realistic
  - - Distance and angles are not always preserved
  - - Parallel lines do not always remain parallel
- Parallel projection
  - + Good for exact measurements
  - + Parallel lines remain parallel
  - - Angles are not (in general) preserved
  - - Less realistic looking

the End

that was enough...