Computer Graphics
Projection

Matthias Teschner

UNI FREIBURG

Homogeneous Coordinates - Summary

- $[x, y, z, w]^\mathsf{T}$ with $w \neq 0$ are the homogeneous coordinates of the 3D position $(\frac{x}{w}, \frac{y}{w}, \frac{z}{w})^\mathsf{T}$
- $[x, y, z, 0]^\mathsf{T}$ is a point at infinity in the direction of $(x, y, z)^\mathsf{T}$
- $[x, y, z, 0]^\mathsf{T}$ is a vector in the direction of $(x, y, z)^\mathsf{T}$
- $\begin{bmatrix} m_{00} & m_{01} & m_{02} & t_0 \\ m_{10} & m_{11} & m_{12} & t_1 \\ m_{20} & m_{21} & m_{22} & t_2 \\ p_0 & p_1 & p_2 & w \end{bmatrix}$ is a transformation that represents rotation, scale, shear, translation, projection

UNI FREIBURG

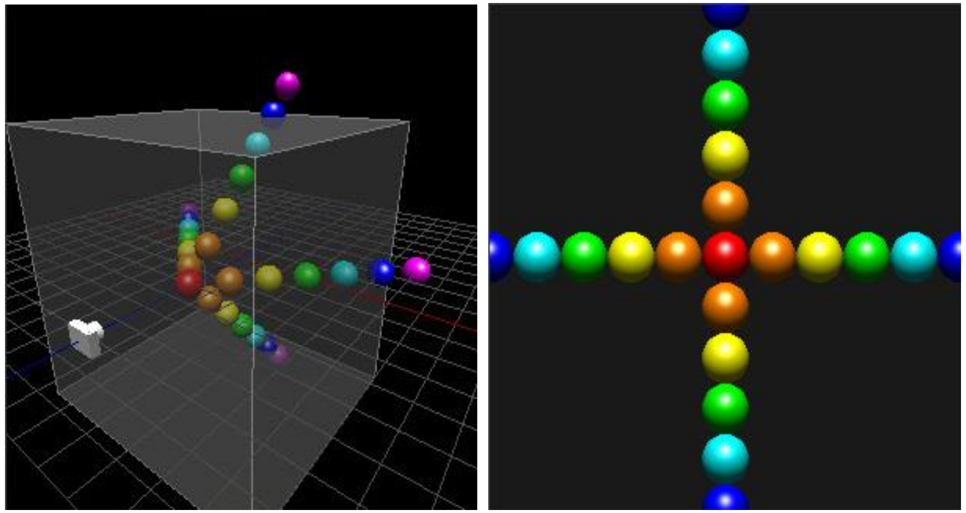
Outline

- Context
- Projections
- Projection transform
- Typical vertex transformations

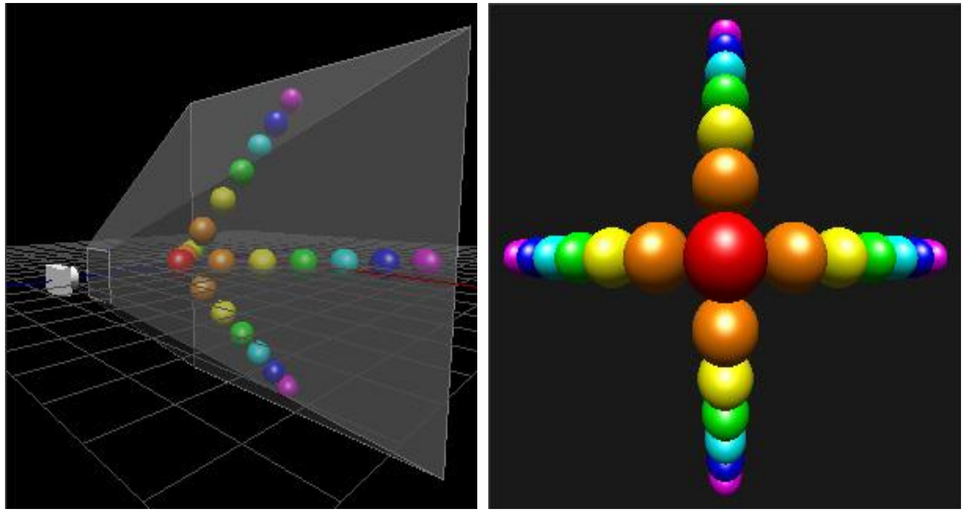
Motivation

– 3D scene with a camera,
its view volume and its projection

[Song Ho Ahn]



Orthographic projection



Perspective projection

Motivation

- Rendering generates planar views from 3D scenes
- 3D space is projected onto a 2D plane considering external and internal camera parameters
  - Position, orientation, focal length
- Projections can be represented with a matrix in homogeneous notation

Motivation

- Transformation matrix in homogeneous notation

$$\begin{bmatrix} m_{00} & m_{01} & m_{02} & t_0 \\ m_{10} & m_{11} & m_{12} & t_1 \\ m_{20} & m_{21} & m_{22} & t_2 \\ p_0 & p_1 & p_2 & w \end{bmatrix}$$

- m_{ij} represent rotation, scale, shear
- t_i represent translation
- p_i are used in projections
- w is the homogeneous component

UNI FREIBURG

Example

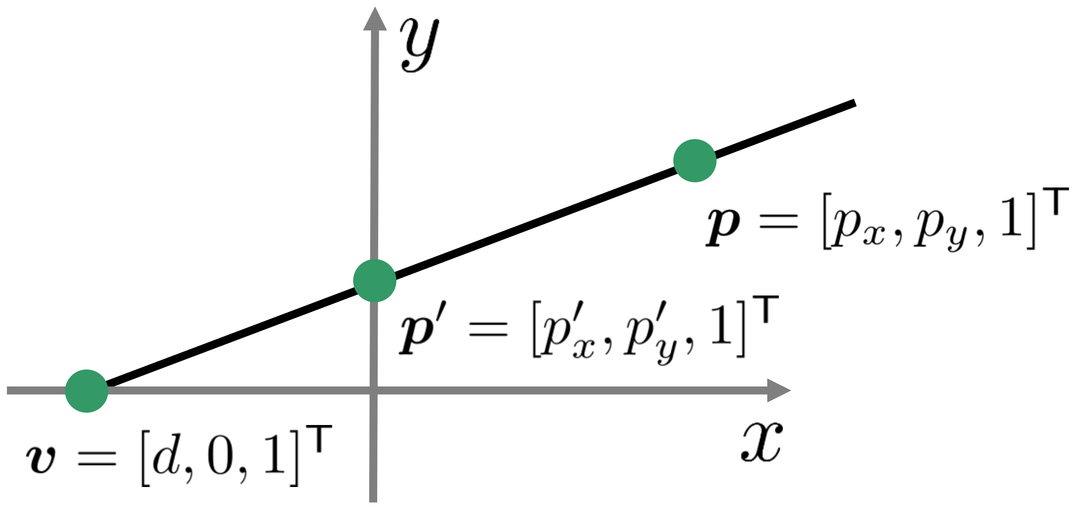
– Last matrix row can be used to realize divisions by a linear combination of multiples of $p_x, p_y, p_z, 1$

$$\boldsymbol{p}' = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ p_0 & p_1 & p_2 & w \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} = \begin{bmatrix} p_x \\ p_y \\ p_z \\ p_0 p_x + p_1 p_y + p_2 p_z + w \end{bmatrix}$$

$$\sim \begin{pmatrix} \frac{p_x}{p_0 p_x + p_1 p_y + p_2 p_z + w} \\ \frac{p_y}{p_0 p_x + p_1 p_y + p_2 p_z + w} \\ \frac{p_z}{p_0 p_x + p_1 p_y + p_2 p_z + w} \end{pmatrix}$$

UNI FREIBURG

2D Illustration



$$p'_x = 0$$

$$\frac{p_y}{p_x - d} = \frac{p'_y}{-d} \quad \Rightarrow \quad p'_y = \frac{-dp_y}{p_x - d}$$

$$\boldsymbol{p}' = \boldsymbol{M}\boldsymbol{p} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -d & 0 \\ 1 & 0 & -d \end{bmatrix} \begin{bmatrix} wp_x \\ wp_y \\ w \end{bmatrix} = \begin{bmatrix} 0 \\ -dwp_y \\ wp_x - wd \end{bmatrix} = \begin{bmatrix} 0 \\ \frac{-dp_y}{p_x - d} \\ 1 \end{bmatrix} \sim \begin{pmatrix} 0 \\ \frac{-dp_y}{p_x - d} \end{pmatrix}$$

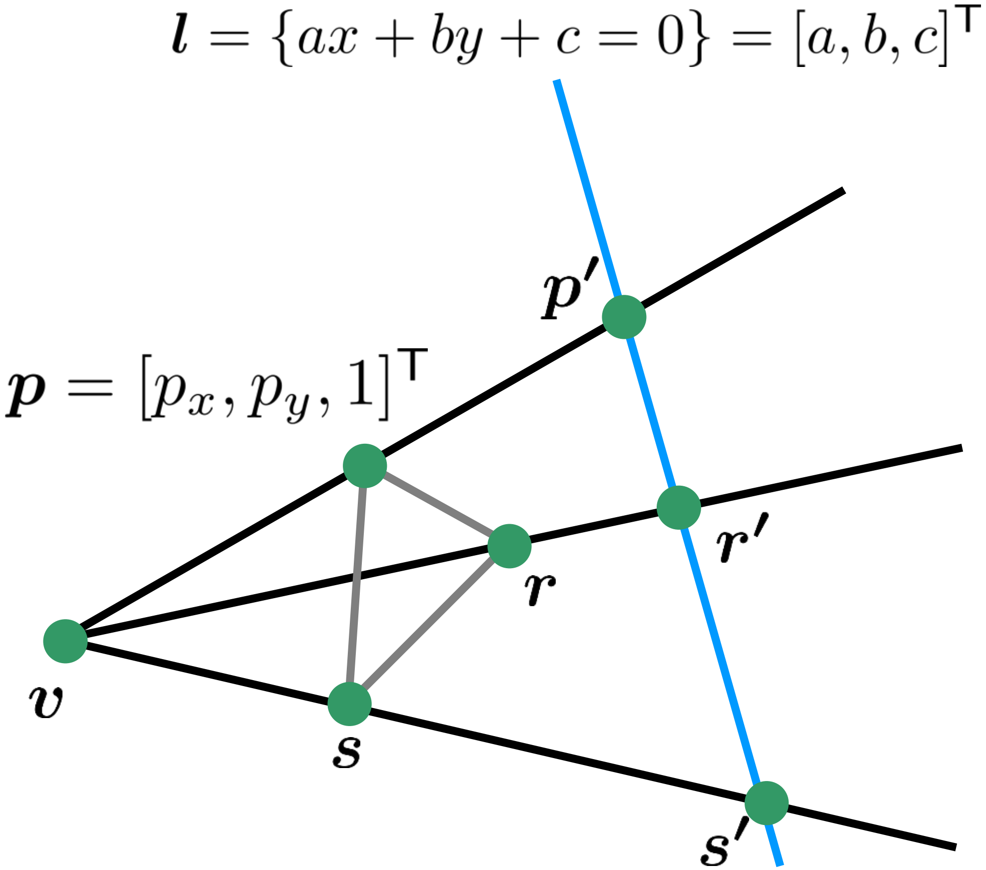
Outline

- Context
- Projections
  - 2D
  - 3D
- Projection transform
- Typical vertex transformations

UNI FREIBURG

Setting

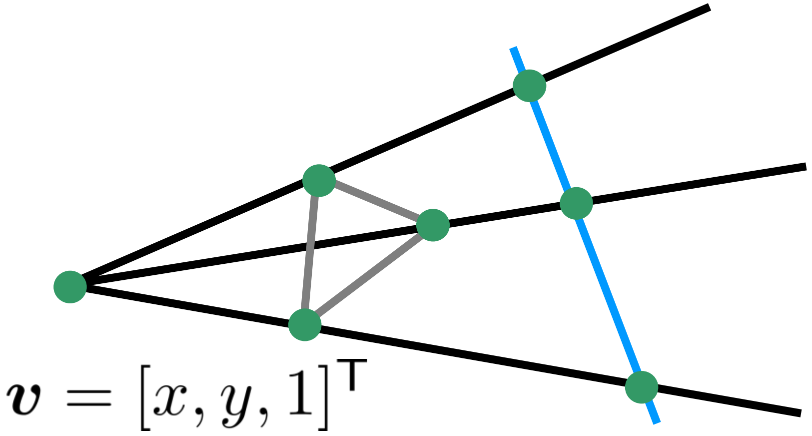
- A 2D projection from v onto l maps a point p onto p'
- p' is the intersection of the line through p and v with line l
- v is the viewpoint, center of perspectivity
- l is the viewline
- The line through p and v is a projector
- v is not on the line l, $p \neq v$

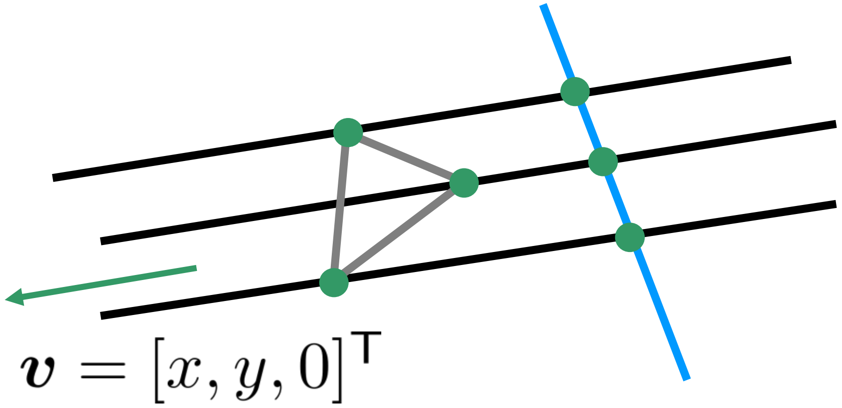


UNI FREIBURG

Classification

- If the homogeneous component of the viewpoint *v* is not equal to zero, we have a perspective projection
  - Projectors are not parallel
- If *v* is at infinity, we have a parallel projection
  - Projectors are parallel





Classification

- Location of viewpoint and orientation of the viewline determine the type of projection
- Parallel (viewpoint at infinity, parallel projectors)
  - Orthographic (viewline orthogonal to the projectors)
  - Oblique (viewline not orthogonal to the projectors)
- Perspective (non-parallel projectors)
  - One-point (viewline intersects one principal axis, i.e. viewline is parallel to a principal axis, one vanishing point)
  - Two-point (viewline intersects two principal axes, two vanishing points)

UNI FREIBURG

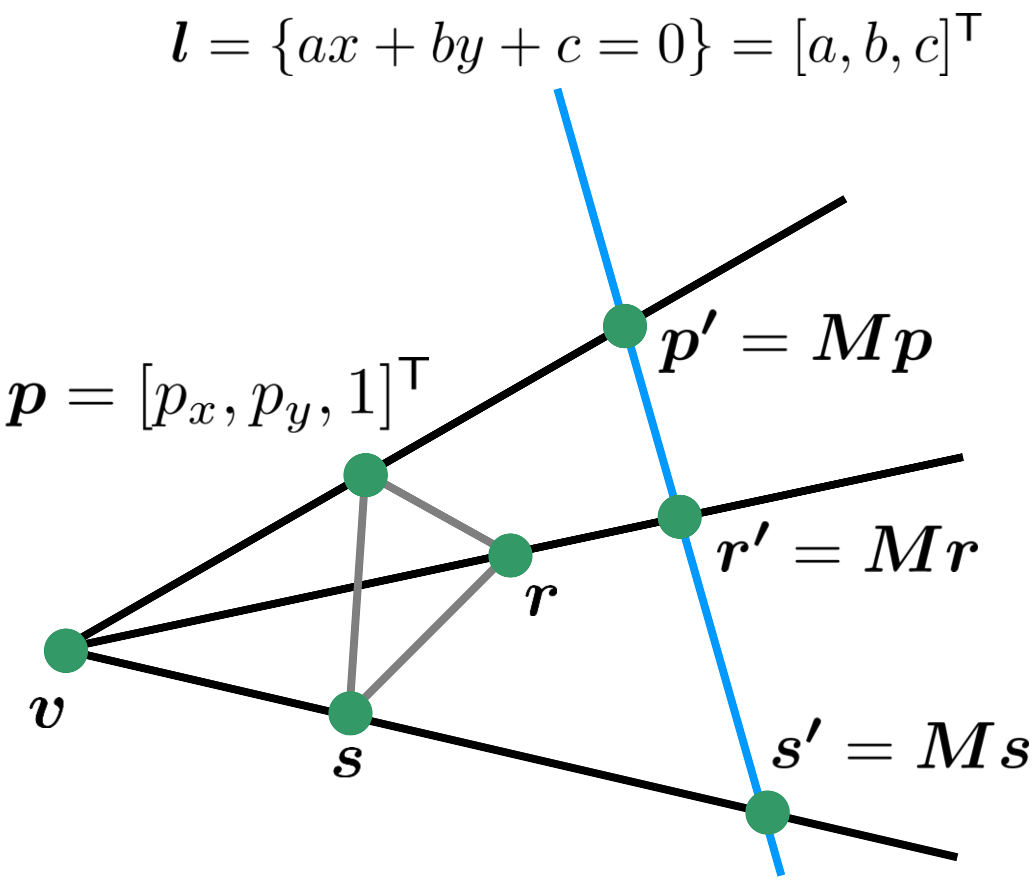
General Case

– A 2D projection is represented by a matrix in homogeneous notation

$$\boldsymbol{M} = \boldsymbol{v}\boldsymbol{l}^\mathsf{T} - (\boldsymbol{l} \cdot \boldsymbol{v})\boldsymbol{I}_3$$

$$\boldsymbol{v}\boldsymbol{l}^\mathsf{T} = \begin{bmatrix} v_x a & v_x b & v_x c \\ v_y a & v_y b & v_y c \\ v_w a & v_w b & v_w c \end{bmatrix}$$

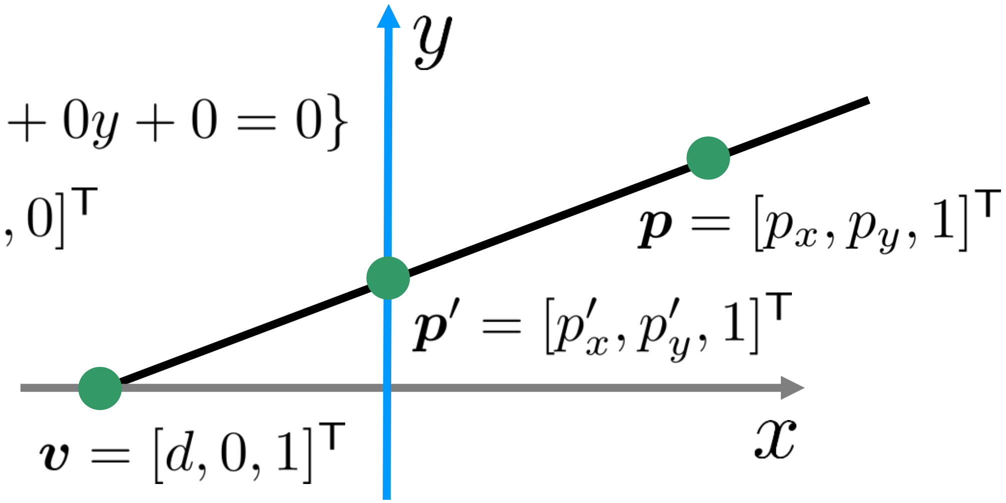
$$(\boldsymbol{l} \cdot \boldsymbol{v})\boldsymbol{I}_3 = (a v_x + b v_y + c v_w) \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



Example

$\boldsymbol{l} = \{1x + 0y + 0 = 0\}$

$\boldsymbol{l} = [1, 0, 0]^\mathsf{T}$



$$\boldsymbol{M} = \begin{bmatrix} d \\ 0 \\ 1 \end{bmatrix} [1, 0, 0] - \left(\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} d \\ 0 \\ 1 \end{bmatrix} \right) \boldsymbol{I}_3$$

$$= \begin{bmatrix} 0 & 0 & 0 \\ 0 & -d & 0 \\ 1 & 0 & -d \end{bmatrix}$$

$$\boldsymbol{p}' = \boldsymbol{M}\boldsymbol{p} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -d & 0 \\ 1 & 0 & -d \end{bmatrix} \begin{bmatrix} wp_x \\ wp_y \\ w \end{bmatrix} = \begin{bmatrix} 0 \\ -dwp_y \\ wp_x - wd \end{bmatrix} = \begin{bmatrix} 0 \\ \frac{-dp_y}{p_x - d} \\ 1 \end{bmatrix} \sim \begin{pmatrix} 0 \\ \frac{-dp_y}{p_x - d} \end{pmatrix}$$

Discussion

– *M* and *λM* represent the same transformation $\lambda \boldsymbol{M}\boldsymbol{p} = \lambda \boldsymbol{p}'$

$\begin{bmatrix} 0 & 0 & 0 \\ 0 & -d & 0 \\ 1 & 0 & -d \end{bmatrix}$ and $\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ -\frac{1}{d} & 0 & 1 \end{bmatrix}$ are the same transformation

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & -d & 0 \\ 1 & 0 & -d \end{bmatrix} \begin{bmatrix} wp_x \\ wp_y \\ w \end{bmatrix} = \begin{bmatrix} 0 \\ -dwp_y \\ wp_x - dw \end{bmatrix} = \begin{bmatrix} 0 \\ \frac{-dp_y}{p_x - d} \\ 1 \end{bmatrix} \sim \begin{pmatrix} 0 \\ \frac{-dp_y}{p_x - d} \end{pmatrix}$$

$$= \begin{pmatrix} 0 \\ \frac{-p_y}{\frac{p_x}{d} - 1} \end{pmatrix} \sim \begin{bmatrix} 0 \\ \frac{-p_y}{\frac{p_x}{d} - 1} \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ wp_y \\ -w\frac{p_x}{d} + w \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ -\frac{1}{d} & 0 & 1 \end{bmatrix} \begin{bmatrix} wp_x \\ wp_y \\ w \end{bmatrix}$$

UNI FREIBURG

Parallel Projection

– Moving *d* to infinity results in parallel projection

$$\lim_{d \to \pm\infty} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ -\frac{1}{d} & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

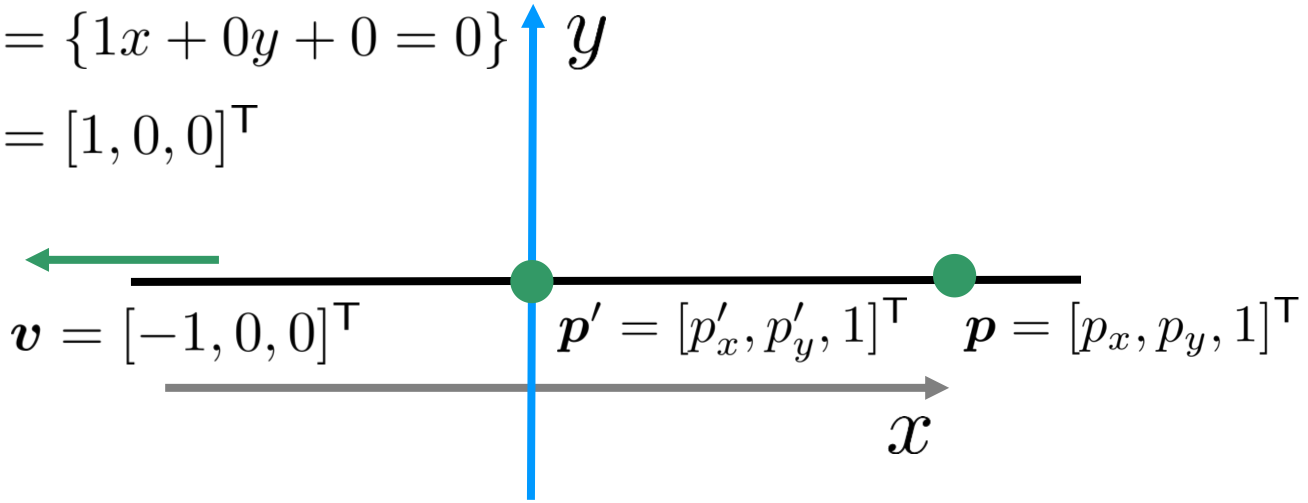
  – *x*-component is mapped to zero
  – *y*- and *w*-component are unchanged

UNI FREIBURG

Parallel Projection

$\boldsymbol{l} = \{1x + 0y + 0 = 0\}$

$\boldsymbol{l} = [1, 0, 0]^\mathsf{T}$



$$\boldsymbol{M} = \boldsymbol{v}\boldsymbol{l}^\mathsf{T} - (\boldsymbol{l} \cdot \boldsymbol{v})\boldsymbol{I}_3$$

$$\boldsymbol{M} = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} (1, 0, 0) - \left(\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} \right) \boldsymbol{I}_3 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

X-component is mapped to zero. *Y*-component is unchanged.

Discussion

- 2D transformation in homogeneous form

$$\boldsymbol{M} = \begin{pmatrix} m_{11} & m_{12} & t_1 \\ m_{21} & m_{22} & t_2 \\ p_1 & p_2 & w \end{pmatrix}$$

- p_1 and p_2 map the homogeneous component w of a point to a value w' that depends on x and y
- Therefore, the scaling of a point depends on x and / or y
- In perspective projections, this is generally employed to scale the x- and y-component with respect to z, its distance to the viewer

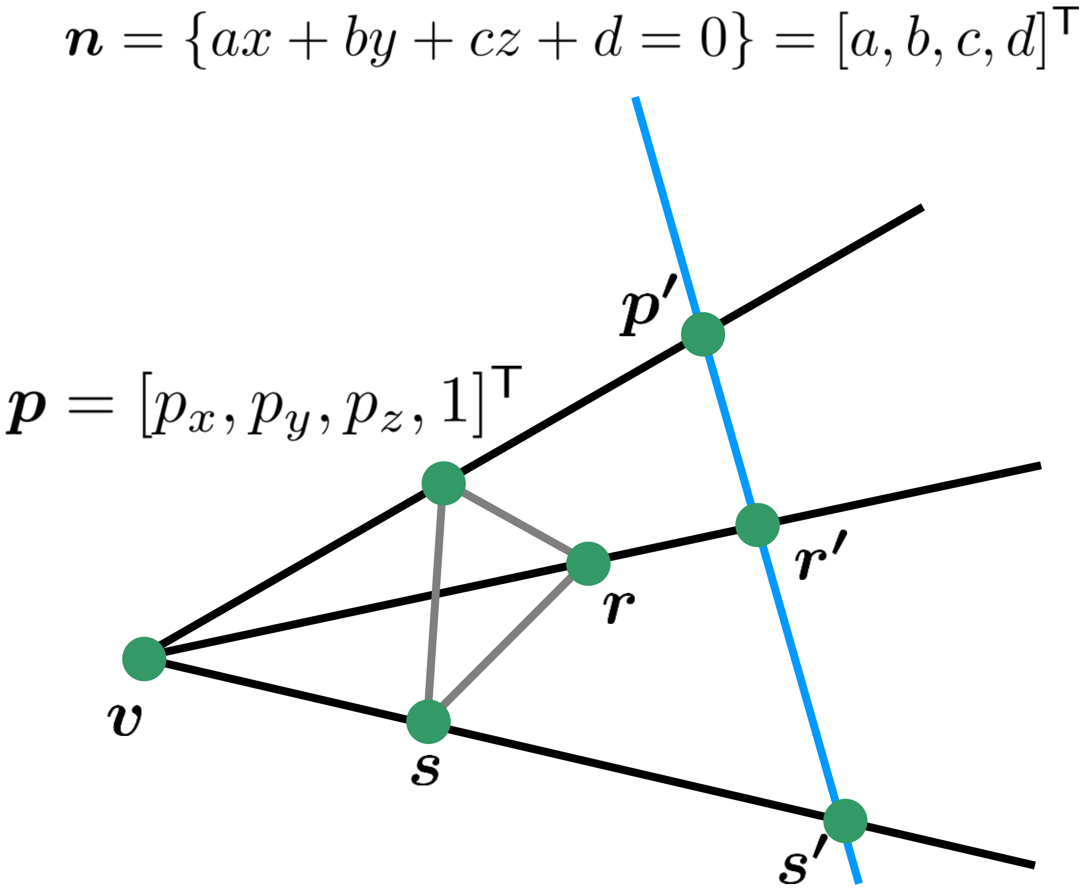
UNI FREIBURG

Outline

- Context
- Projections
  - 2D
  - 3D
- Projection transform
- Typical vertex transformations

Setting

- A 3D projection from *v* onto *l* maps a point *p* onto *p*'
- *p*' is the intersection of the line through *p* and *v* with plane *n*
- *v* is the viewpoint, center of perspectivity
- *n* is the viewplane
- The line through *p* and *v* is a projector
- *v* is not on the plane *n*, $p \neq v$



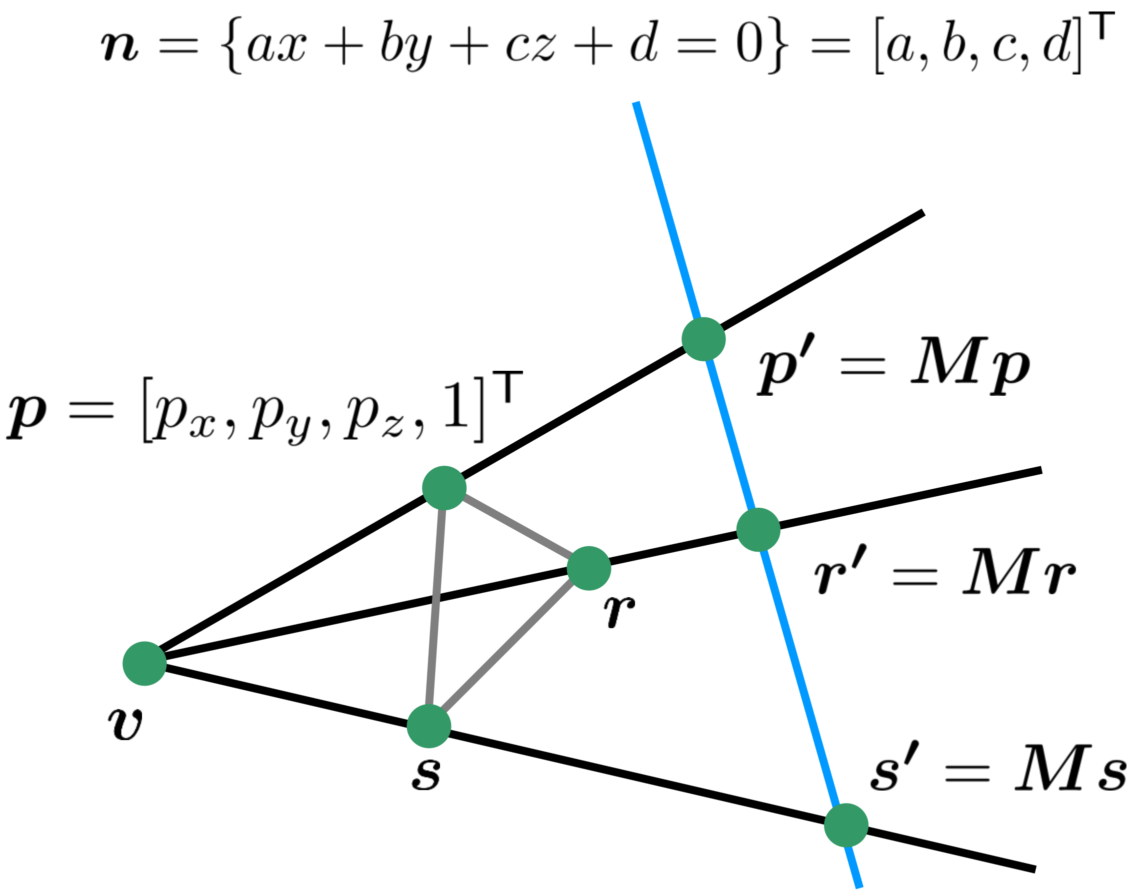
General Case

– A 3D projection is represented by a matrix in homogeneous notation

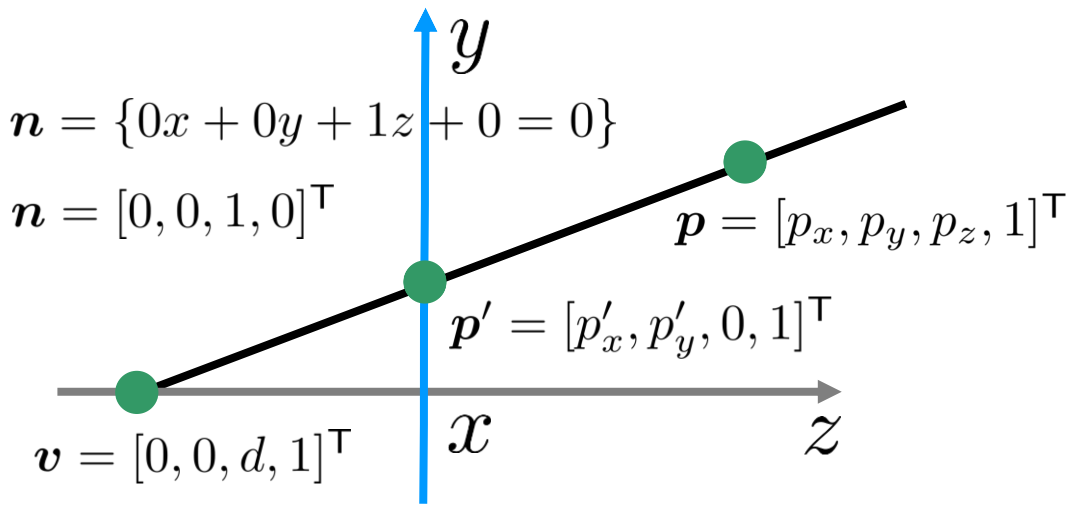
$$\boldsymbol{M} = \boldsymbol{v}\boldsymbol{n}^\mathsf{T} - (\boldsymbol{n} \cdot \boldsymbol{v})\boldsymbol{I}_4$$

$$\boldsymbol{v}\boldsymbol{n}^\mathsf{T} = \begin{bmatrix} v_x a & v_x b & v_x c & v_x d \\ v_y a & v_y b & v_y c & v_y d \\ v_z a & v_z b & v_z c & v_z d \\ v_w a & v_w b & v_w c & v_w d \end{bmatrix}$$

$$(\boldsymbol{n} \cdot \boldsymbol{v})\boldsymbol{I}_4 = (av_x + bv_y + cv_z + dv_w)\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Example



$$\frac{p'_x}{-d} = \frac{p_x}{p_z - d}$$

$$\frac{p'_y}{-d} = \frac{p_y}{p_z - d}$$

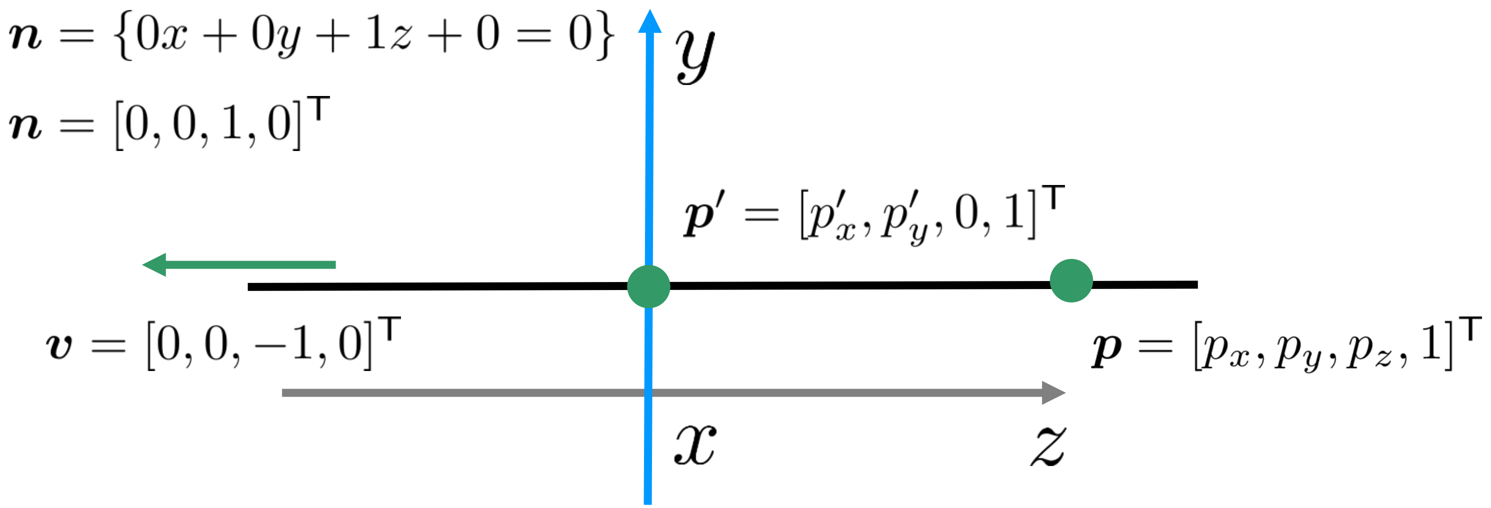
$$p'_z = 0$$

$$\boldsymbol{M} = \begin{bmatrix} 0 \\ 0 \\ d \\ 1 \end{bmatrix} (0, 0, 1, 0) - \left(\begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ d \\ 1 \end{bmatrix} \right) \boldsymbol{I}_4$$

$$= \begin{bmatrix} -d & 0 & 0 & 0 \\ 0 & -d & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -d \end{bmatrix}$$

$$\boldsymbol{p}' = \boldsymbol{M}\boldsymbol{p} = \begin{bmatrix} -d & 0 & 0 & 0 \\ 0 & -d & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -d \end{bmatrix} \begin{bmatrix} wp_x \\ wp_y \\ wp_z \\ w \end{bmatrix} = \begin{bmatrix} -dwp_x \\ -dwp_y \\ 0 \\ wp_z - dw \end{bmatrix} = \begin{bmatrix} \frac{-dp_x}{p_z - d} \\ \frac{-dp_y}{p_z - d} \\ 0 \\ 1 \end{bmatrix} \sim \begin{pmatrix} \frac{-dp_x}{p_z - d} \\ \frac{-dp_y}{p_z - d} \\ 0 \end{pmatrix}$$

Parallel Projection



$$\boldsymbol{M} = \boldsymbol{v}\boldsymbol{n}^\mathsf{T} - (\boldsymbol{n} \cdot \boldsymbol{v})\boldsymbol{I}_4$$

$$\boldsymbol{M} = \begin{bmatrix} 0 \\ 0 \\ -1 \\ 0 \end{bmatrix} [0, 0, 1, 0] - \left(\begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ -1 \\ 0 \end{bmatrix} \right) \boldsymbol{I}_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

X- and *y*-component are unchanged. *Z*-component is mapped to zero.

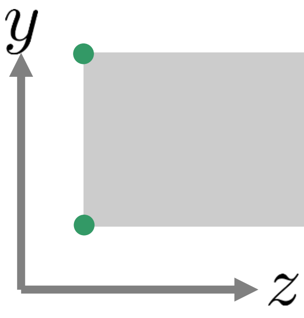
Outline

- Context
- Projections
- Projection transform
  - Motivation
  - Perspective projection
  - Discussion
  - Orthographic projection
- Typical vertex transformations

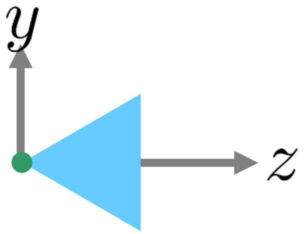
UNI FREIBURG

Modelview Transform

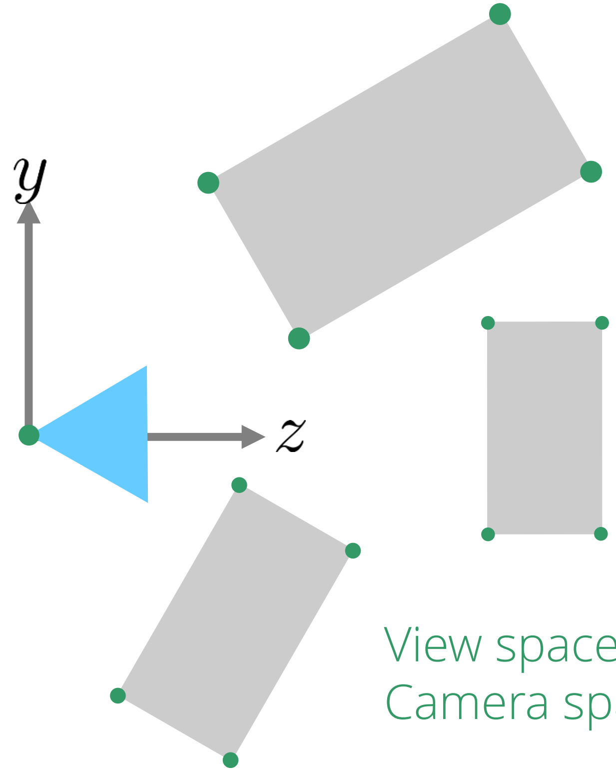
Local coordinate system of an object



Local coordinate system of a camera

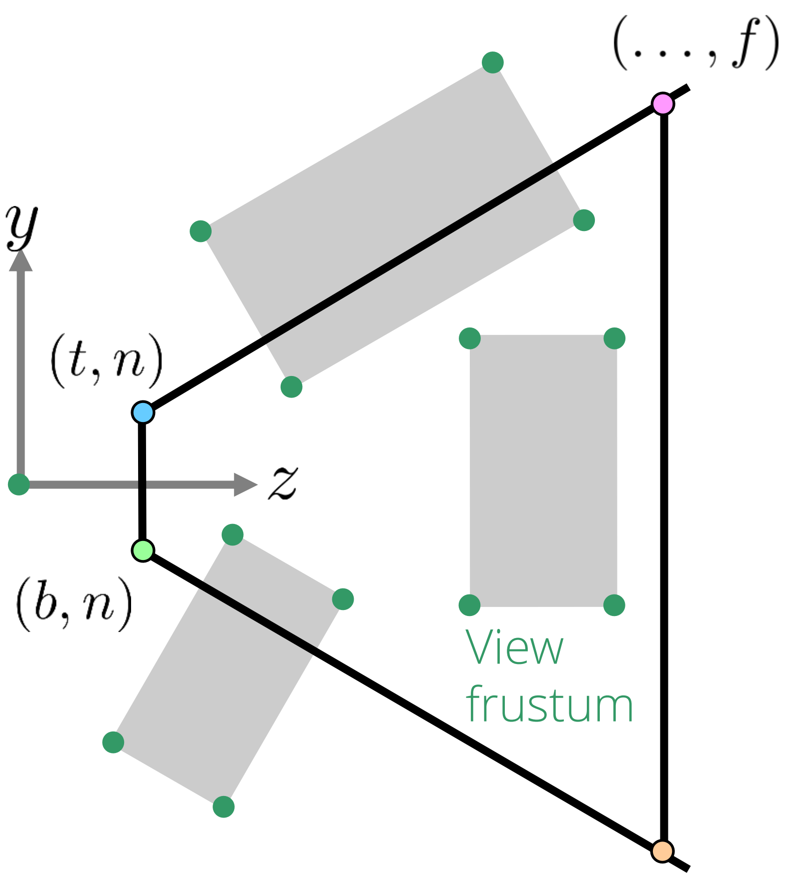


Transformation from local into view space is realized with the modelview transform.
Objects: $V^{-1}M_1$, $V^{-1}M_2$, $V^{-1}M_3$
Camera: $V^{-1}V = I$



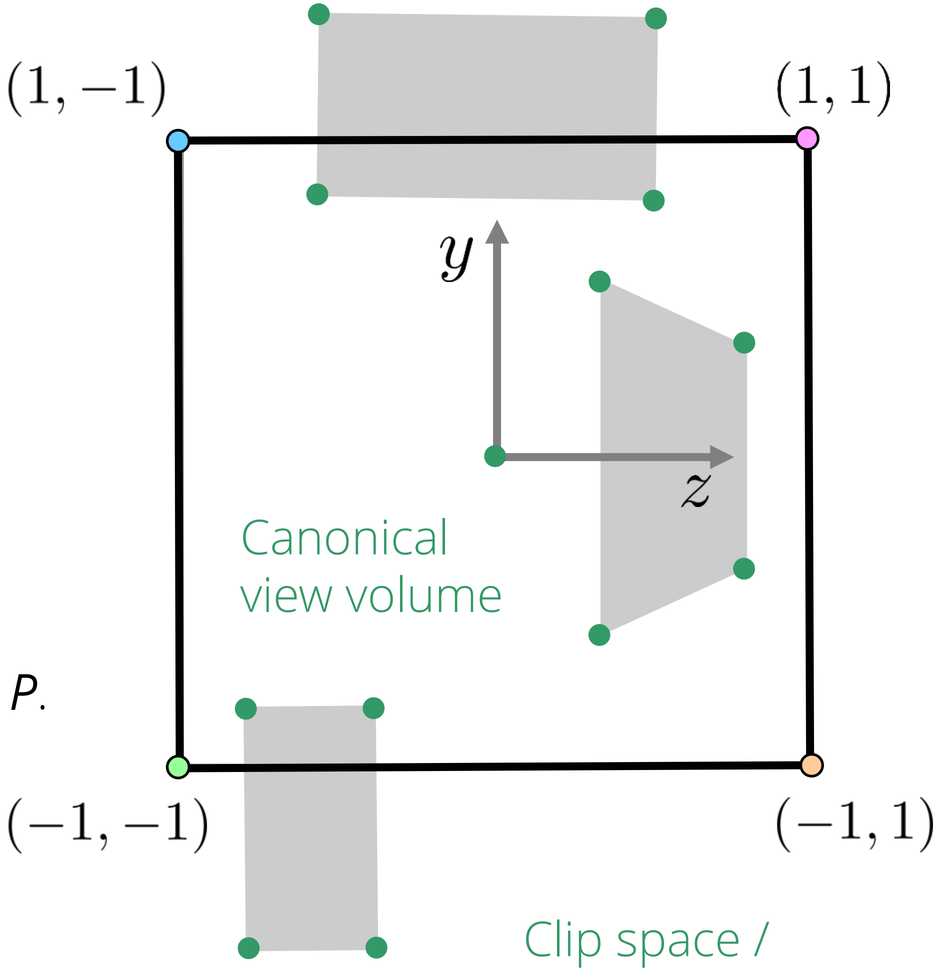
View space / Camera space.

Projection Transform



View space / Camera space.

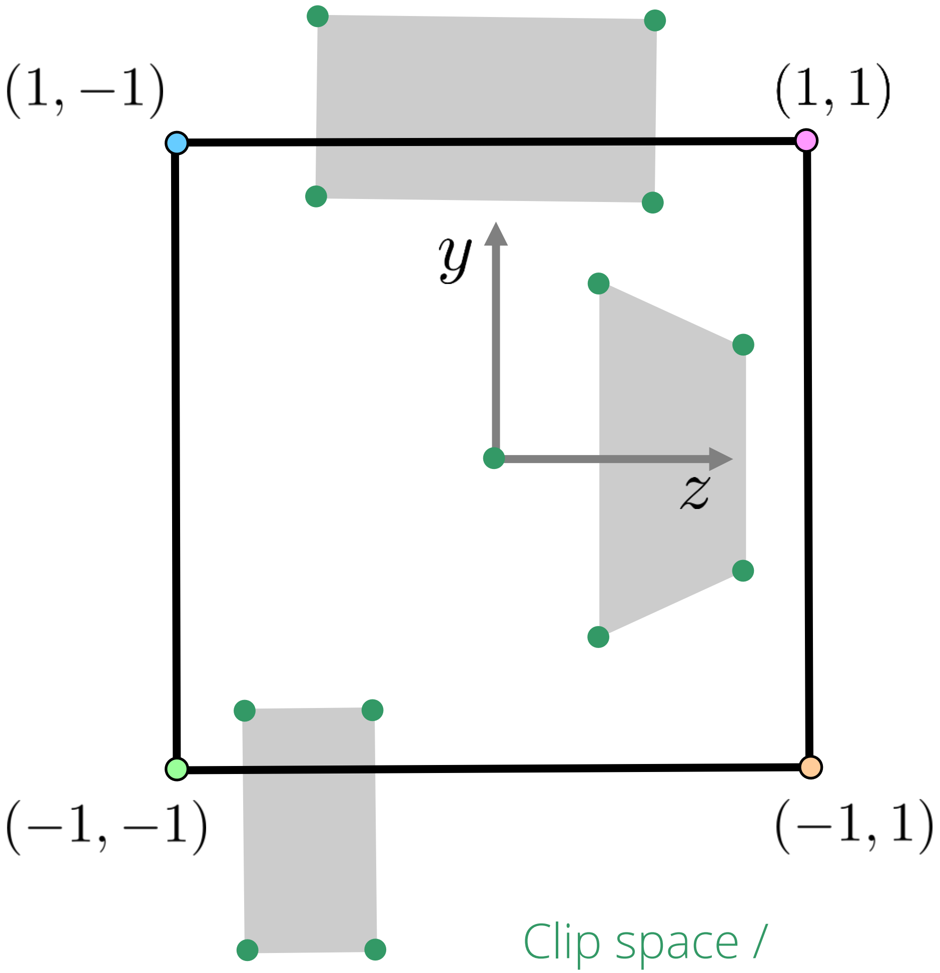
Transformation from view space to clip space / NDC space is realized with the projection transform *P*.



Clip space / NDC space.

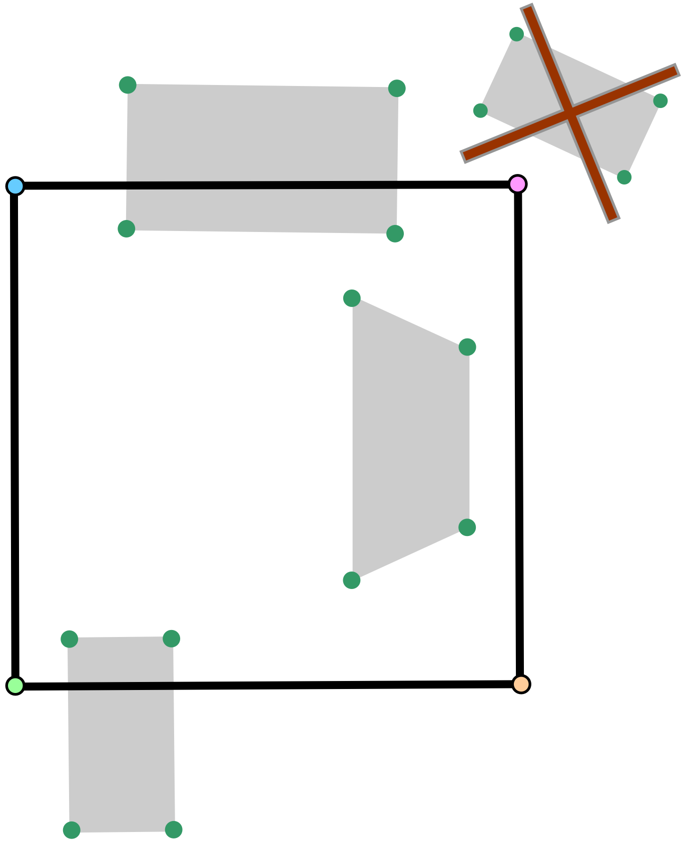
Clip Space / NDC Space

- Allows simplified and unified implementations
  - Culling
  - Clipping
  - Visibility
    - Parallel ray casting
    - Depth test
  - Projection onto view plane / screen (viewport mapping)

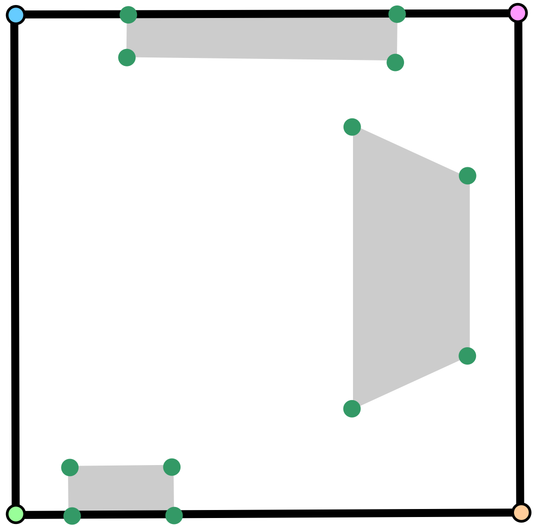


Clip space / NDC space.

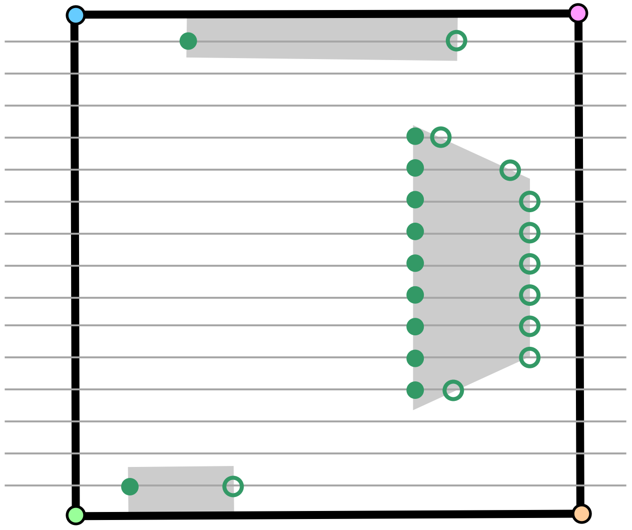
Culling / Clipping / Visibility



Culling



Clipping



Visibility

Outline

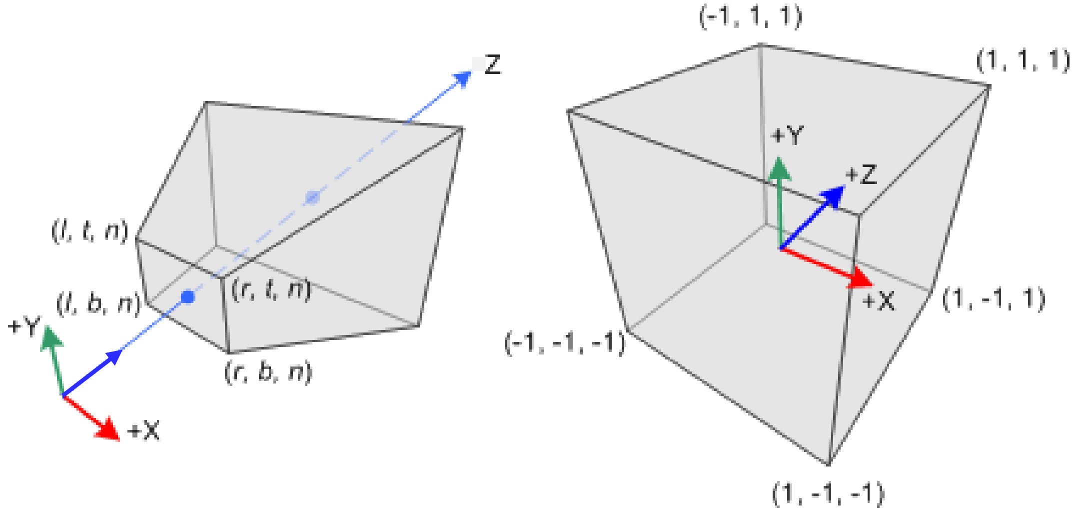
- Context
- Projections
- Projection transform
  - Motivation
  - Perspective projection
  - Discussion
  - Orthographic projection
- Typical vertex transformations

UNI FREIBURG

Perspective Projection Transform

- Maps a view volume / pyramidal frustum to a canonical view volume
- The view volume is specified by its boundary
  - Left *l*, right *r*, bottom *b*, top *t*, near *n*, far *f*
- The canonical view volume is, e.g., a cube from (-1,-1,-1) to (1,1,1)

[Song Ho Ahn]



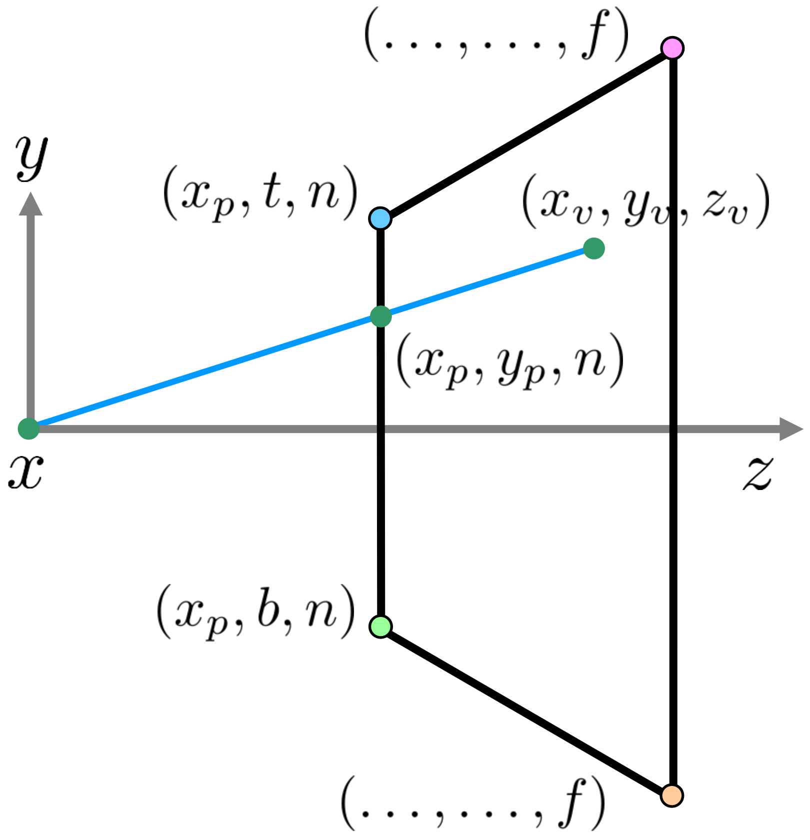
UNI FREIBURG

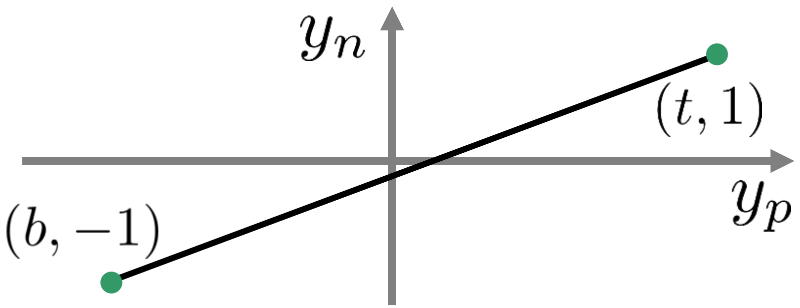
Perspective Projection Transform

- Is applied to vertices
- Maps
  - The *x*-component of a projected point from (left, right) to (-1, 1)
  - The *y*-component of a projected point from (bottom, top) to (-1, 1)
  - The *z*-component of a point from (near, far) to (-1, 1)
- If a point in view space is inside / outside the view volume, it is inside /outside the canonical view volume

UNI FREIBURG

Derivation





$$\frac{y_p}{n} = \frac{y_v}{z_v} \Rightarrow y_p = \frac{n y_v}{z_v} \qquad x_p = \frac{n x_v}{z_v}$$

$$y_n = \alpha y_p + \beta$$

$$\alpha = \frac{1-(-1)}{t-b} \qquad \beta = -\frac{t+b}{t-b}$$

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

$$y_n = \frac{1}{z_v}\left(\frac{2n}{t-b} y_v - \frac{t+b}{t-b} z_v\right)$$

$$x_n = \frac{1}{z_v}\left(\frac{2n}{r-l} x_v - \frac{r+l}{r-l} z_v\right)$$

Derivation

– From

$$x_n = \frac{1}{z_v}\left(\frac{2n}{r-l}x_v - \frac{r+l}{r-l}z_v\right) \qquad y_n = \frac{1}{z_v}\left(\frac{2n}{t-b}y_v - \frac{t+b}{t-b}z_v\right)$$

we get

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ w_c \end{bmatrix} = \begin{bmatrix} \frac{2n}{r-l} & 0 & -\frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & -\frac{t+b}{t-b} & 0 \\ \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix}$$

Clip coordinates
(clip space)

with

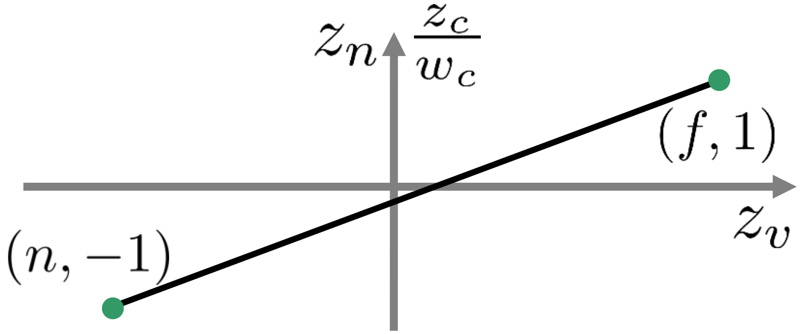
$$\begin{bmatrix} x_n \\ y_n \\ z_n \\ 1 \end{bmatrix} = \begin{bmatrix} x_c/w_c \\ y_c/w_c \\ z_c/w_c \\ w_c/w_c \end{bmatrix}$$

Normalized device
coordinates
(NDC space)

Derivation

- z_v is mapped from (near, far) or (n, f) to (-1, 1)
- The transform does not depend on x_v and y_v
- So, we have to solve for A and B in

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ w_c \end{bmatrix} = \begin{bmatrix} \frac{2n}{r-l} & 0 & -\frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & -\frac{t+b}{t-b} & 0 \\ 0 & 0 & A & B \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} w_v x_v \\ w_v y_v \\ w_v z_v \\ w_v \end{bmatrix}$$



$$z_n = \frac{z_c}{w_c} = \frac{Az_v + Bw_v}{z_v}$$

Derivation

– z_v=n with w_v=1 is mapped to z_n=-1

– z_v=f with w_v=1 is mapped to z_n=1

$$\Rightarrow A = \frac{f+n}{f-n} \qquad \Rightarrow B = -\frac{2fn}{f-n}$$

– The complete projection matrix is

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

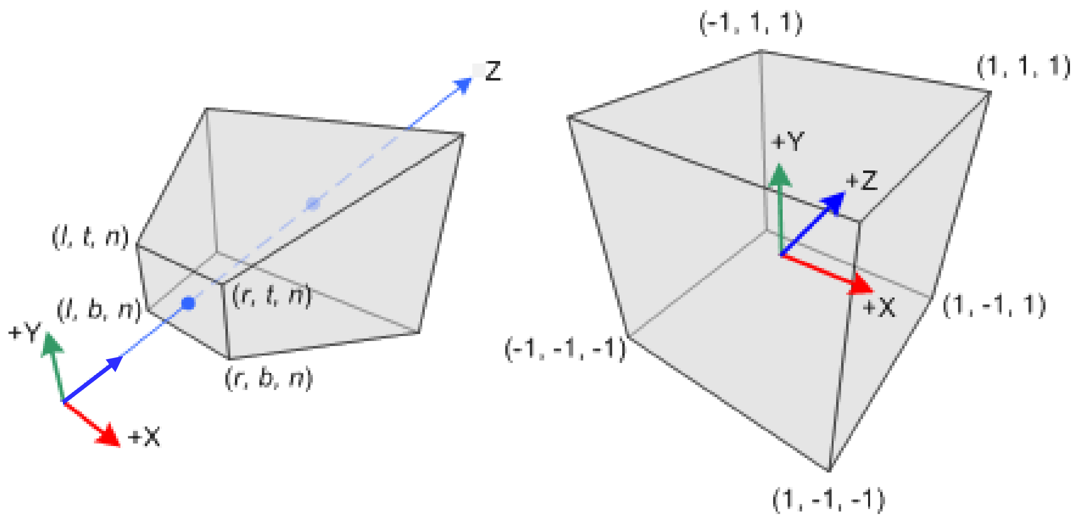
UNI FREIBURG

Perspective Projection Matrix

$$\boldsymbol{P} = \begin{bmatrix} \frac{2n}{r-l} & 0 & -\frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & -\frac{t+b}{t-b} & 0 \\ 0 & 0 & \frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

transforms the view volume, the pyramidal frustum to the canonical view volume

[Song Ho Ahn]



Outline

- Context
- Projections
- Projection transform
  - Motivation
  - Perspective projection
  - Discussion
  - Orthographic projection
- Typical vertex transformations

Symmetric Setting

– The matrix simplifies for *r=-l* and *t=-b*

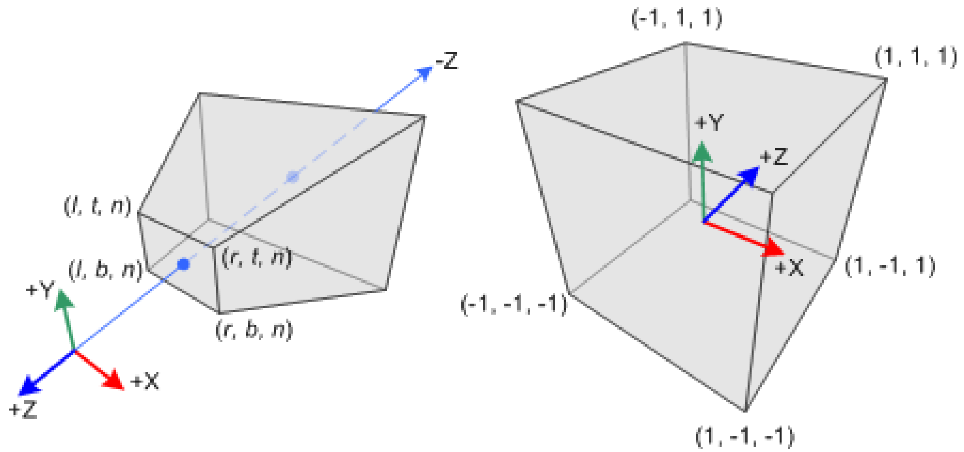
$$\begin{aligned} r+l&=0\\ r-l&=2r\\ t+b&=0\\ t-b&=2t \end{aligned} \quad \Rightarrow \boldsymbol{P}=\begin{bmatrix} \frac{n}{r} & 0 & 0 & 0\\ 0 & \frac{n}{t} & 0 & 0\\ 0 & 0 & \frac{f+n}{f-n} & -\frac{2fn}{f-n}\\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Variants

- Projection matrices depend on coordinate systems and other settings
- E.g., OpenGL
  - Viewing along negative *z*-axis in view space
  - Negated values for *n* and *f*

$$\boldsymbol{P} = \begin{bmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

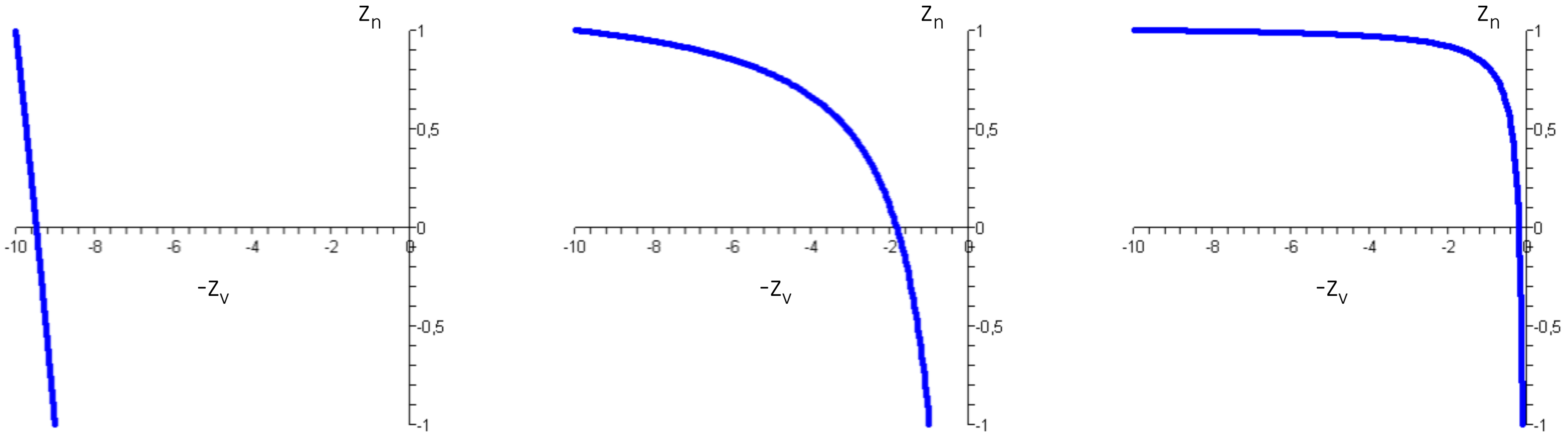
[Song Ho Ahn]



Non-linear Mapping of Depth Values

$$z_n = \frac{f+n}{f-n} - \frac{1}{z_v}\frac{2fn}{f-n}$$

– Near plane should not be too close to zero



$n = 9 \quad f = 10$ $\qquad$ $n = 1 \quad f = 10$ $\qquad$ $n = 0.1 \quad f = 10$

UNI FREIBURG

Non-linear Mapping of Depth Values

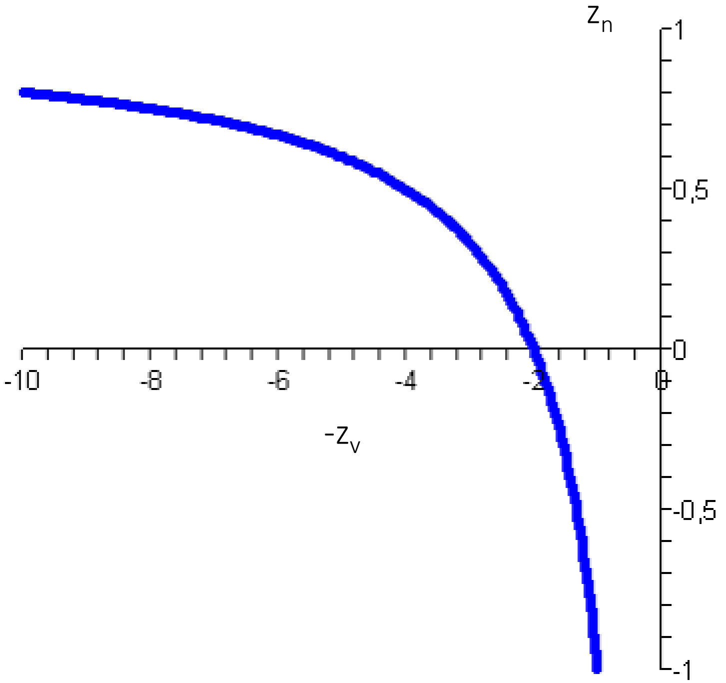
– Setting the far plane to infinity is not too critical

$$\boldsymbol{P} = \begin{bmatrix} \frac{2n}{r-l} & 0 & -\frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & -\frac{t+b}{t-b} & 0 \\ 0 & 0 & \frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$f \rightarrow \infty$$

$$\Rightarrow \begin{bmatrix} \frac{2n}{r-l} & 0 & -\frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & -\frac{t+b}{t-b} & 0 \\ 0 & 0 & 1 & -2n \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\Rightarrow z_n = 1 - \frac{2n}{z_v}$$



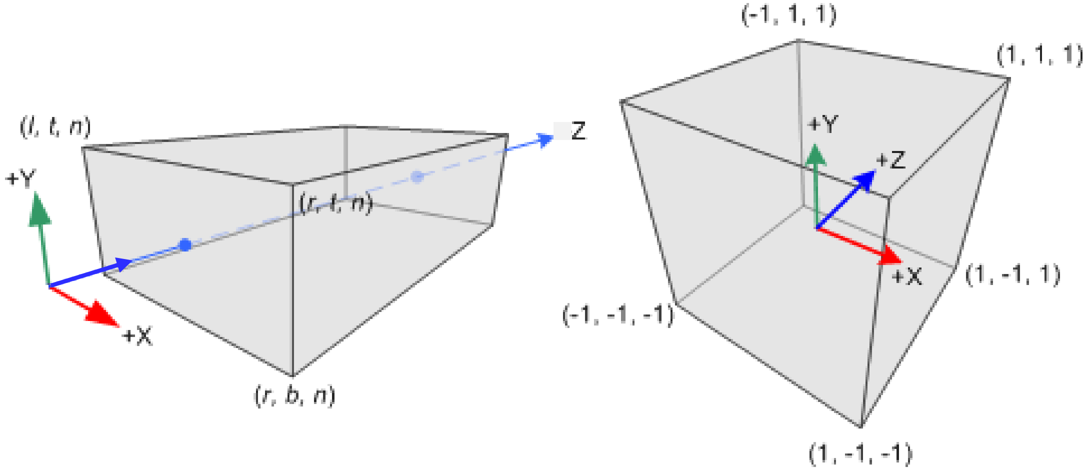
$n = 1 \quad f = \infty$

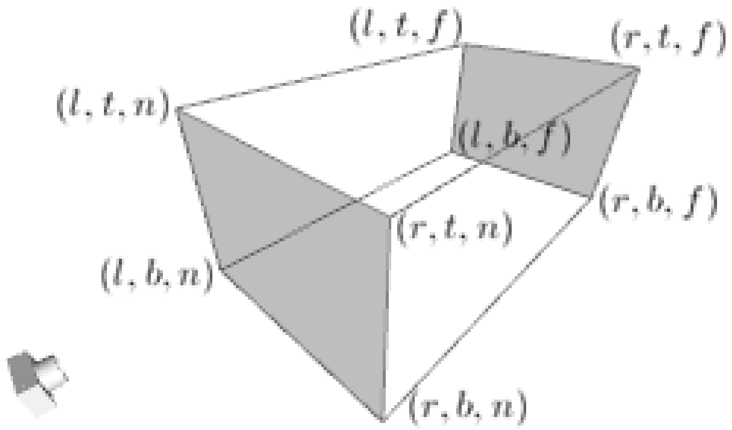
Outline

- Context
- Projections
- Projection transform
  - Motivation
  - Perspective projection
  - Discussion
  - Orthographic projection
- Typical vertex transformations

Orthographic Projection

- View volume is a cuboid and specified by its boundary
  - Left *l*, right *r*, bottom *b*, top *t*, near *n*, far *f*
- Canonical view volume is a cube from (-1,-1,-1) to (1,1,1)

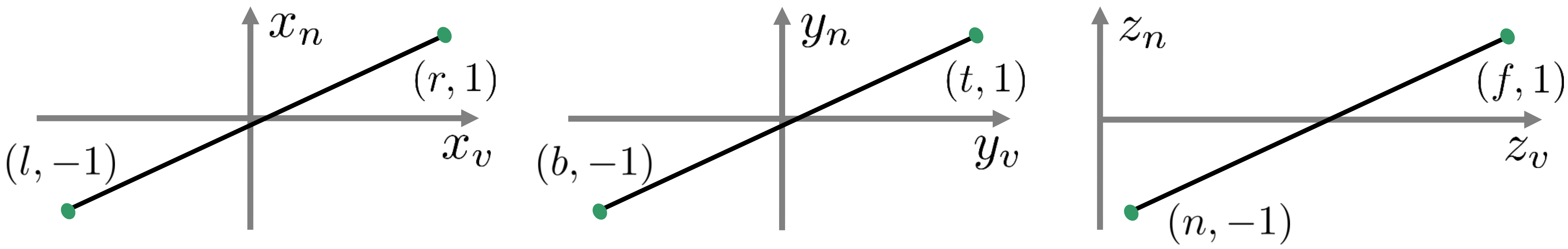




[Song Ho Ahn]

Derivation

– All components of a point in view coordinates are linearly mapped to the range of (-1,1)



$$x_n = \frac{2}{r-l} x_v - \frac{r+l}{r-l} \qquad y_n = \frac{2}{t-b} y_v - \frac{t+b}{t-b} \qquad z_n = \frac{2}{f-n} z_v - \frac{f+n}{f-n}$$

– Linear in x_v, y_v, z_v

– Combination of scale and translation

UNI FREIBURG

Orthographic Projection Matrix

– General form

$$\boldsymbol{P} = \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & \frac{2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

– Simplified form for a symmetric view volume

$$r + l = 0$$
$$r - l = 2r$$
$$t + b = 0$$
$$t - b = 2t$$

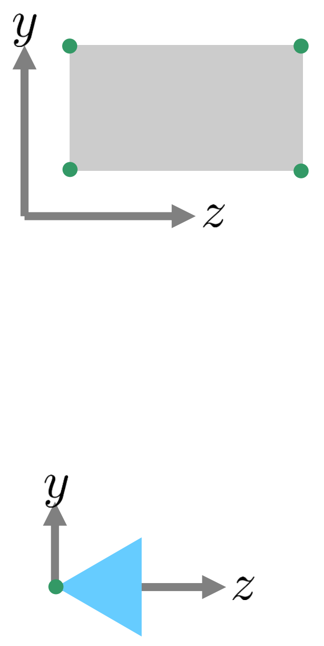
$$\Rightarrow \boldsymbol{P} = \begin{bmatrix} \frac{1}{r} & 0 & 0 & 0 \\ 0 & \frac{1}{t} & 0 & 0 \\ 0 & 0 & \frac{2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

UNI FREIBURG

Outline

- Context
- Projections
- Projection transform
- Typical vertex transformations

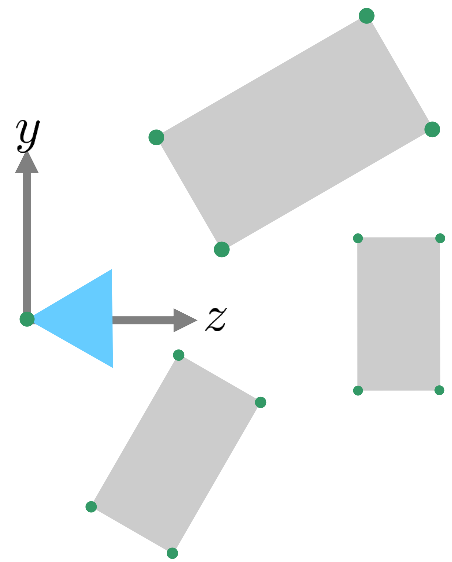
Overview



Local space

$$\boldsymbol{V}^{-1}\boldsymbol{M}_i$$

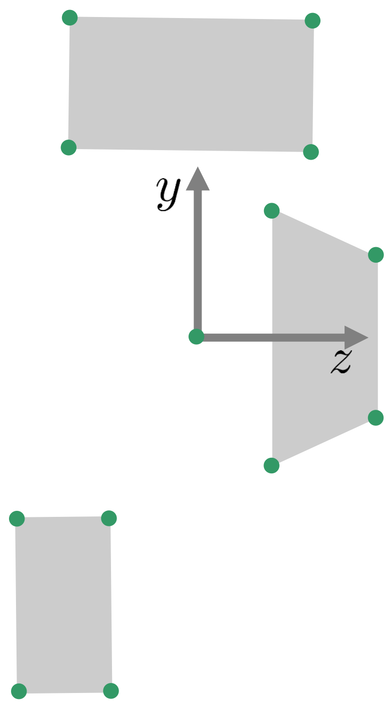
Modelview transform depends on model *i*.



View space

$$\boldsymbol{P}$$

Projection transform depends on camera parameters.



Clip space

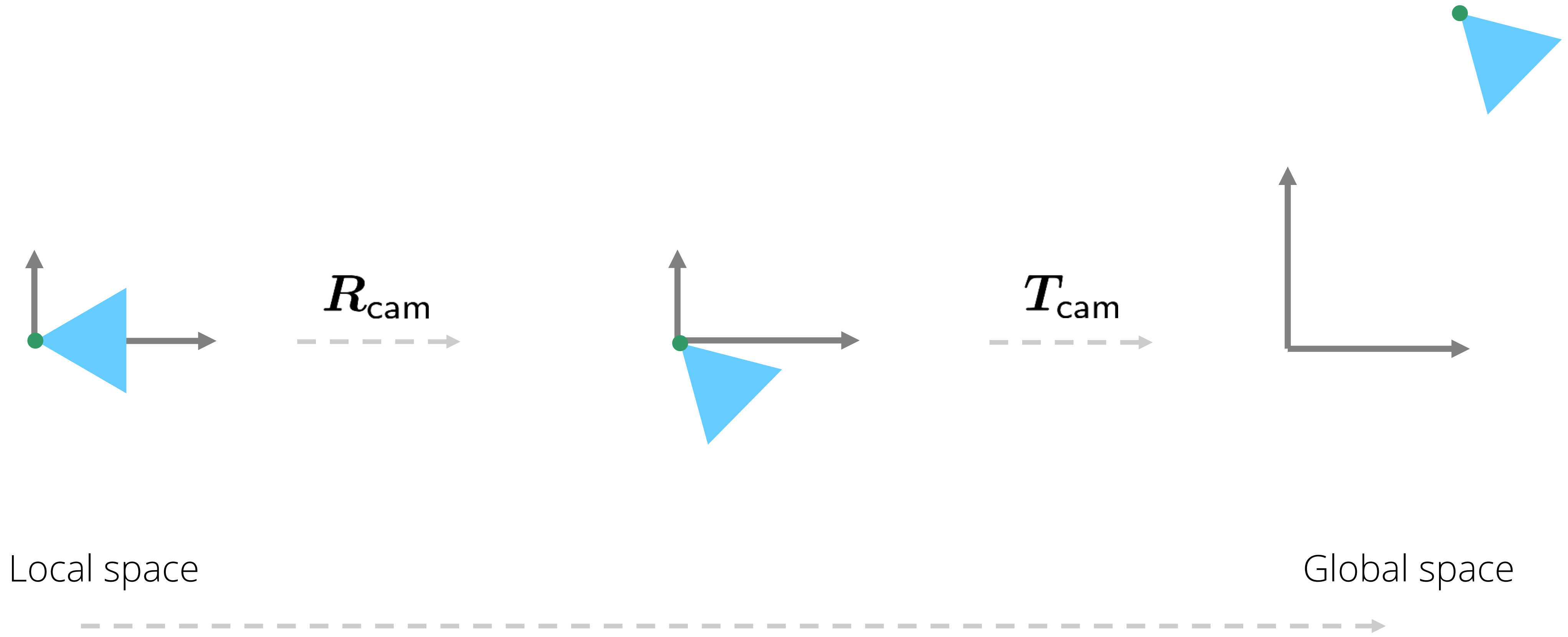
$$\boldsymbol{P}\boldsymbol{V}^{-1}\boldsymbol{M}_i$$

Coordinate Systems

Model transform:	Local space	⇨	Global space
View transform:	Local space	⇨	Global space
Inverse view transform:	Global space	⇨	View space
Modelview transform:	Local space	⇨	View space
Projection transform:	View space	⇨	Clip space

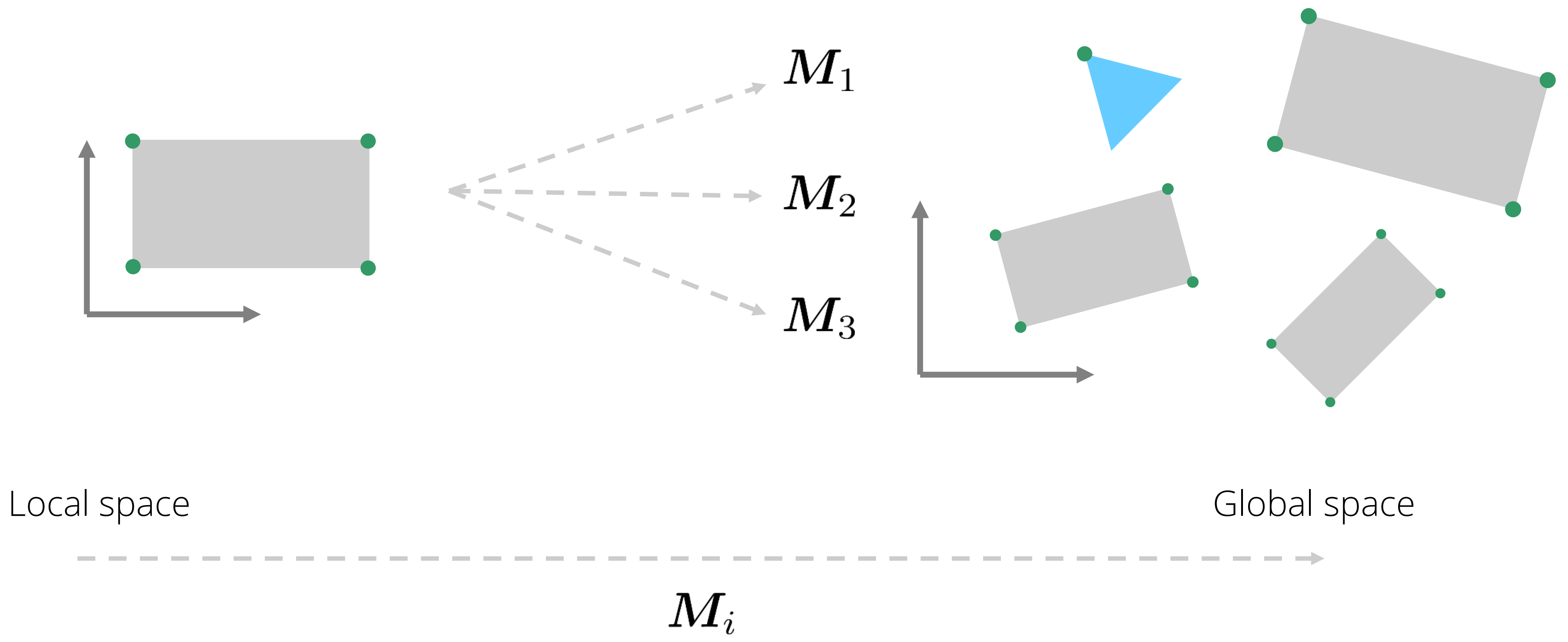
UNI FREIBURG

Camera Placement

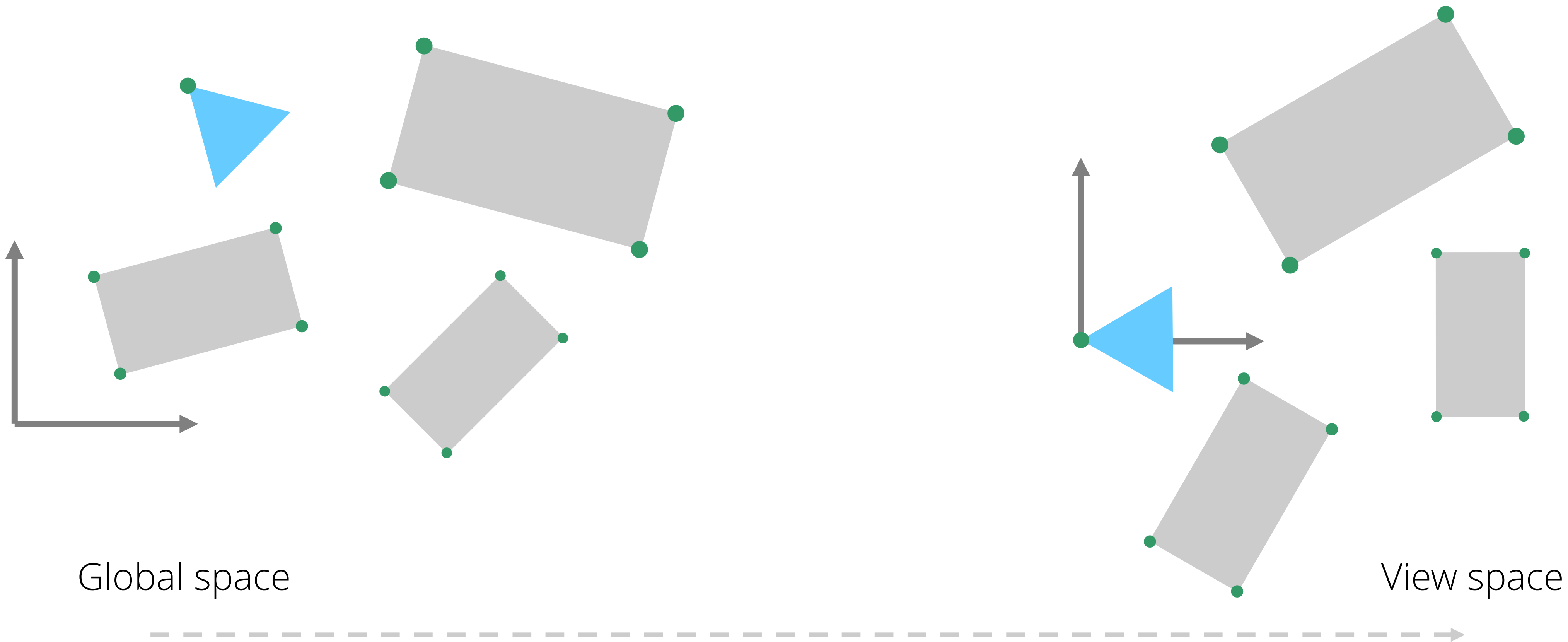


$$V = T_{\text{cam}} R_{\text{cam}}$$

Object Placement

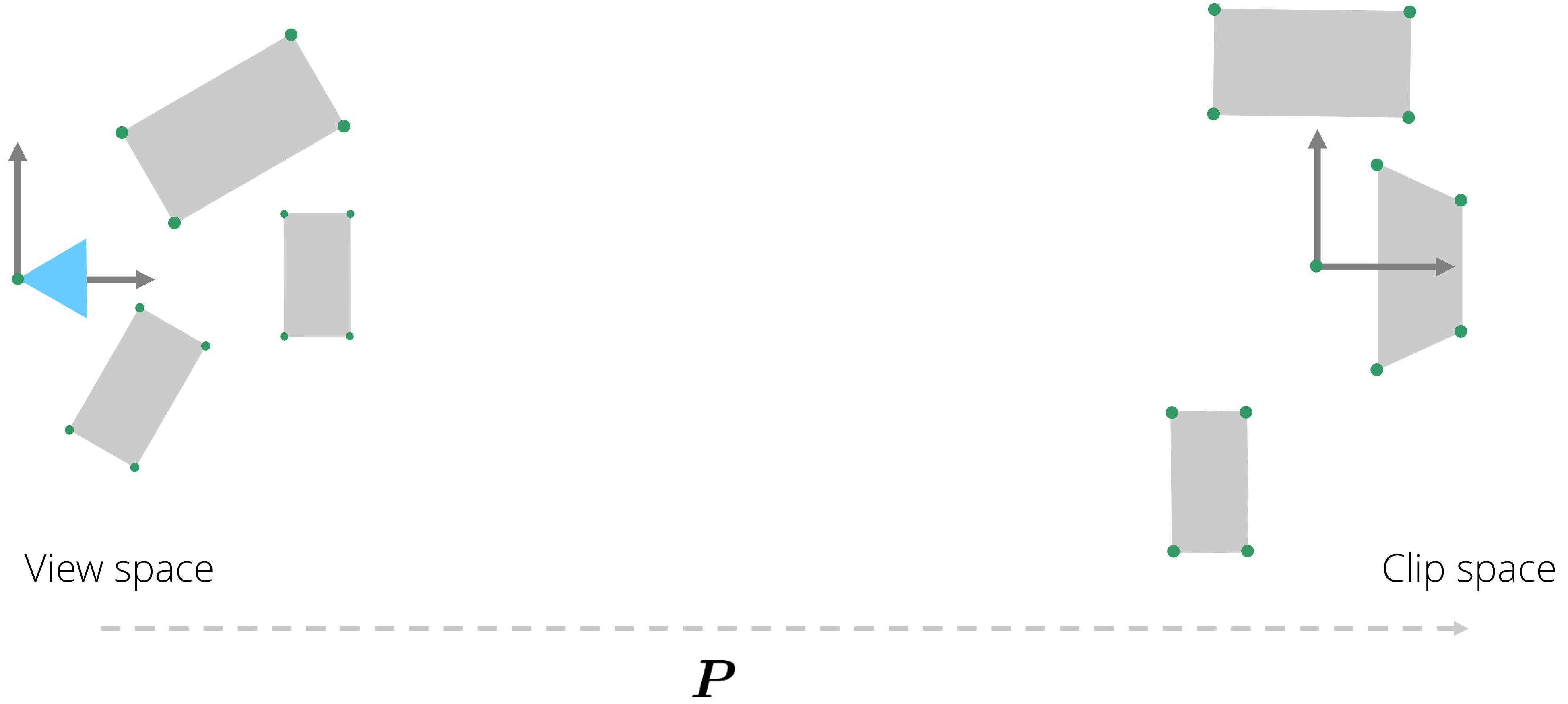


View Transform

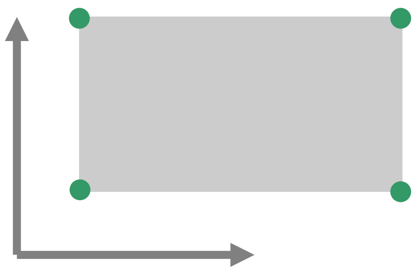


$$V^{-1} = (T_{\text{cam}} R_{\text{cam}})^{-1} = R_{\text{cam}}^{-1} T_{\text{cam}}^{-1} = R_{\text{cam}}^{\mathsf{T}} T_{\text{cam}}^{-1}$$

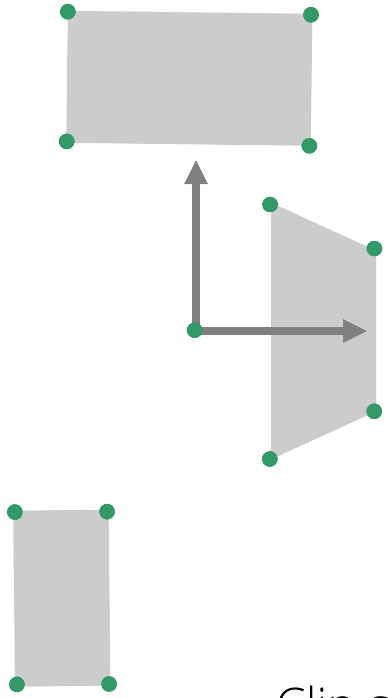
Projection Transform



Vertex Transforms - Summary



Transformations are applied to vertices. Internal and external camera parameters are encoded in the matrices for view and projection transform.



Local space

Clip space

$$\boldsymbol{P}\boldsymbol{R}^{\mathsf{T}}_{\text{cam}}\boldsymbol{T}^{-1}_{\text{cam}}\boldsymbol{M}_i$$

References

– Song Ho Ahn: "OpenGL", http://www.songho.ca/ .
– Duncan Marsh: "Applied Geometry for Computer Graphics and CAD", Springer Verlag, Berlin, 2004.

UNI FREIBURG