Computer Graphics Homogeneous Notation

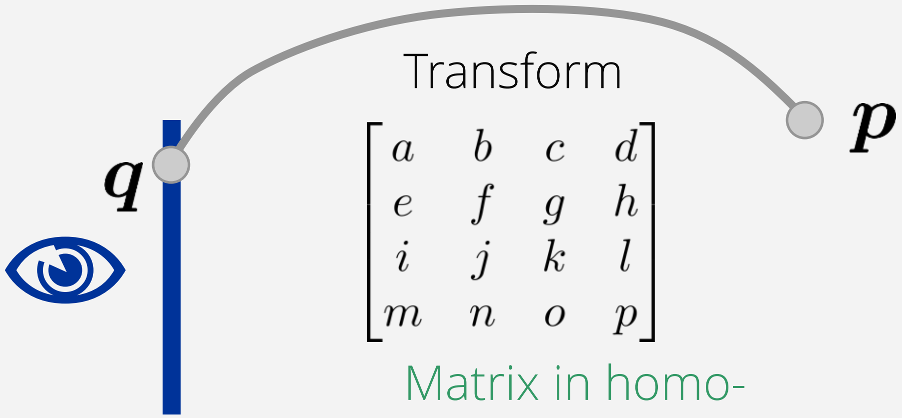
Matthias Teschner

UNI FREIBURG

What is visible at the sensor?

– Visibility can be resolved by ray casting or by applying transformations

Ray Casting computes ray-scene intersections to estimate *q* from *p*.



Matrix in homo-geneous notation

Rasterizers apply transformations to *p* in order to estimate *q*. *p* is projected onto the sensor plane.

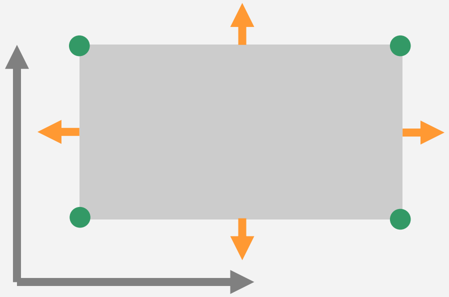
Outline

- Motivation
- Homogeneous notation
- Transformations

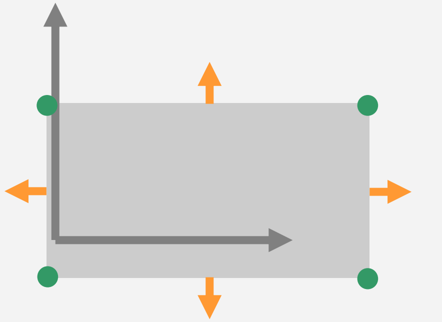
Motivation

- Transformations in modeling and rendering
  - Position, reshape, and animate objects, lights, cameras
  - Project 3D geometry onto the camera plane
- Homogeneous notation
  - 3D vertices (positions) and 3D normals (directions) are represented with 4D vectors
  - Transformations are represented with 4x4 matrices
  - All transformations of positions and directions are consistently realized as a matrix-vector product

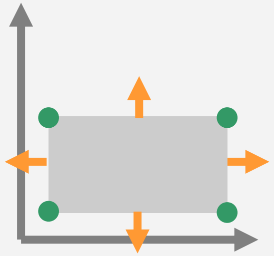
Transformations – 2D



Four faces / primitives / polygons, four points / vertices, four normals.

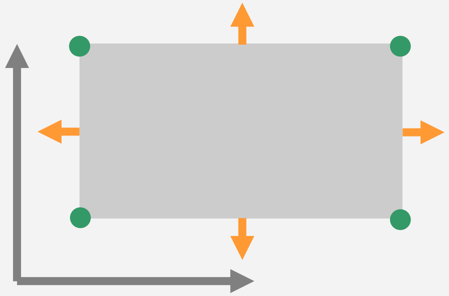


Translation.

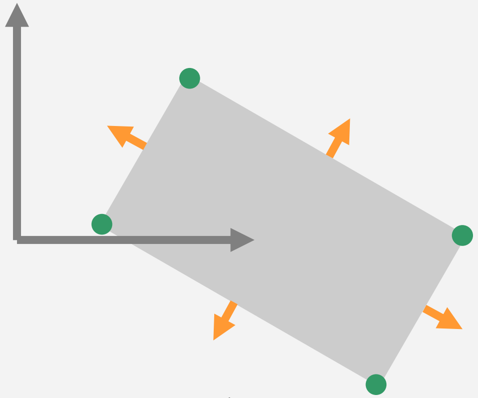


Scale.

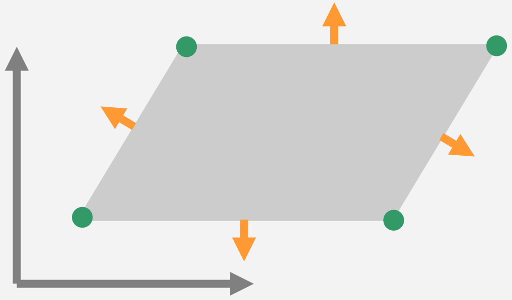
Transformations change vertex positions and surface normals.



Identity transform.



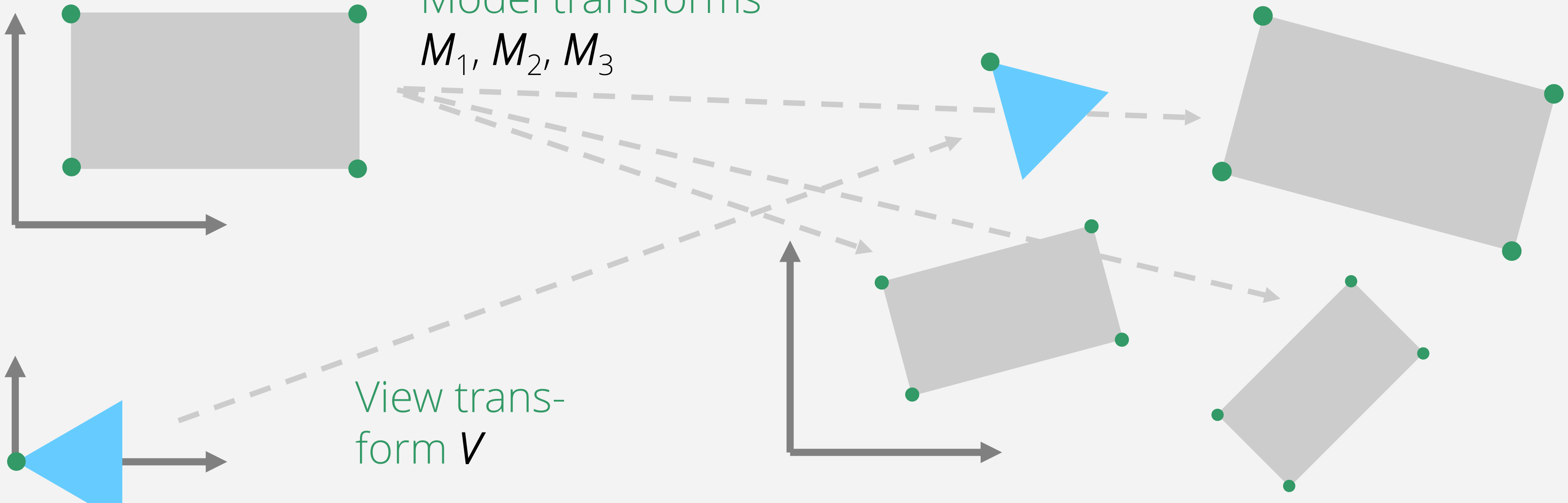
Rotation.



Shear.

Coordinate Systems and Transformations

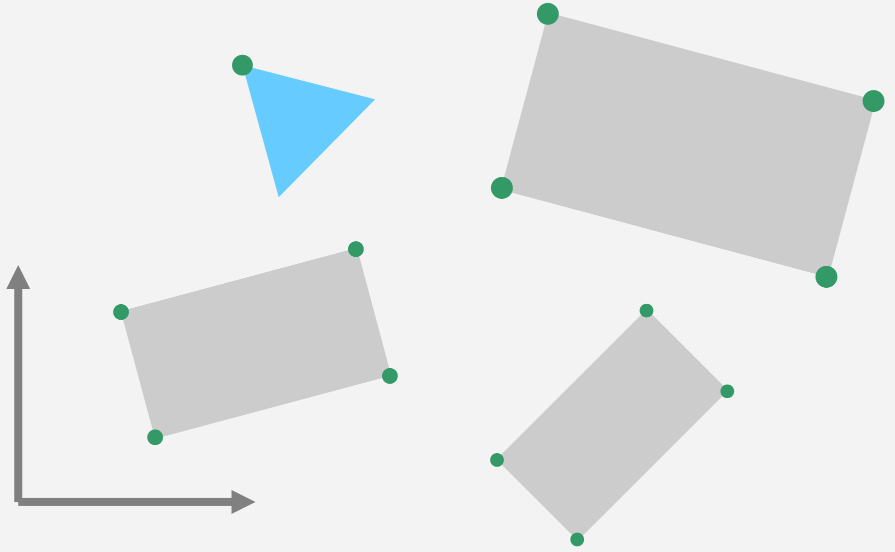
Local coordi-
nate system
of an object



Local coordi-
nate system
of a camera

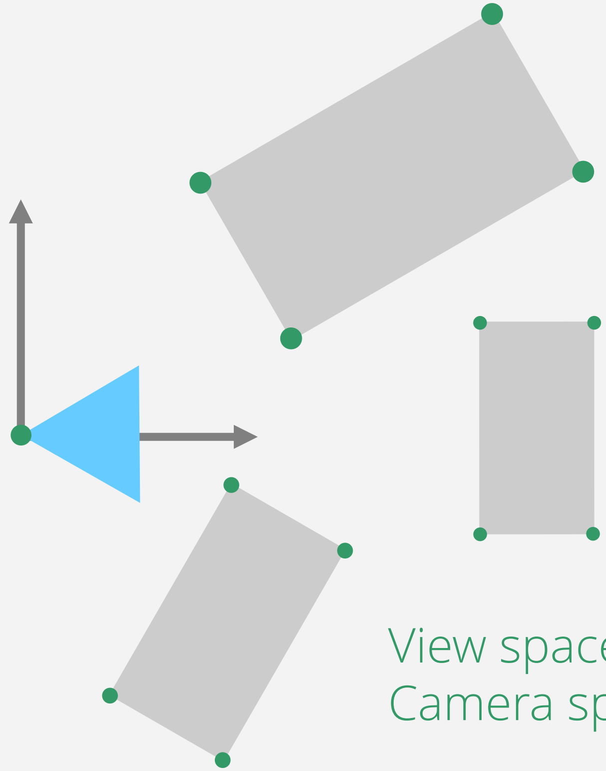
Global coordinate system with one camera
and three instances of the same object

Coordinate Systems and Transformations



Global coordinate system with one camera and three objects

Inverse view transform V^{-1} applied to all objects and the camera

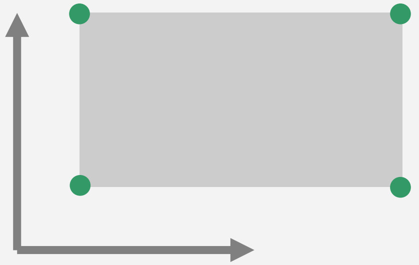


View space / Camera space.

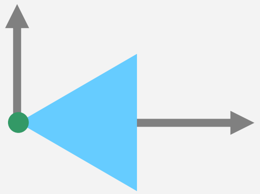
Working in view space is motivated by simplified implementations. E.g., rays start at **0** in view space.

Modelview Transform

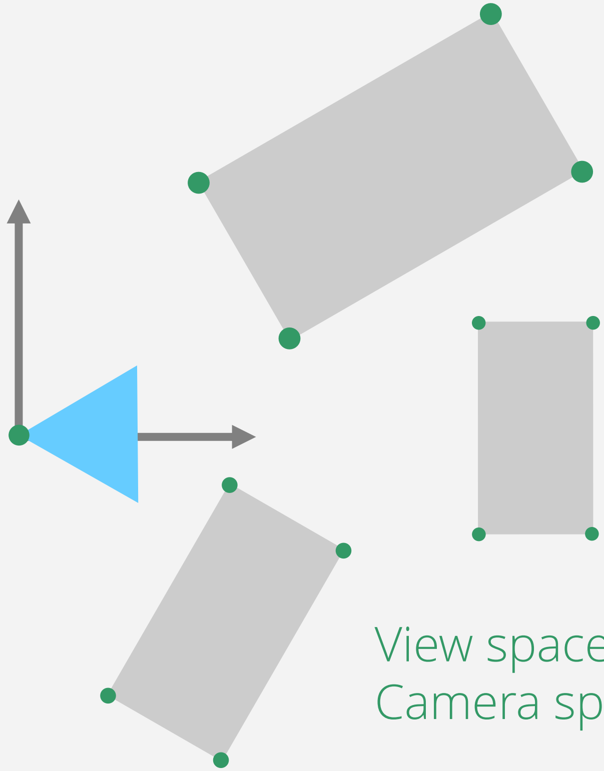
Local coordinate system of an object



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.

More Transformations

- To transform from view space positions to positions on the camera plane
  - Projection transform
  - Viewport transform
- See lecture on projections

Transformations - Groups

- Translation, rotation, reflection
  - Preserve shape and size
  - Congruent transformations (Euclidean transformations)
- Translation, rotation, reflection, scale
  - Preserve shape
  - Similarity transformations

Affine Transformations

- Translation, rotation, reflection, scale, shear
  - Angles and lengths are not preserved
  - Preserve collinearity
    - Points on a line are transformed to points on a line
  - Preserve proportions
    - Ratios of distances between points are preserved
  - Preserve parallelism
    - Parallel lines are transformed to parallel lines

Affine Transformations

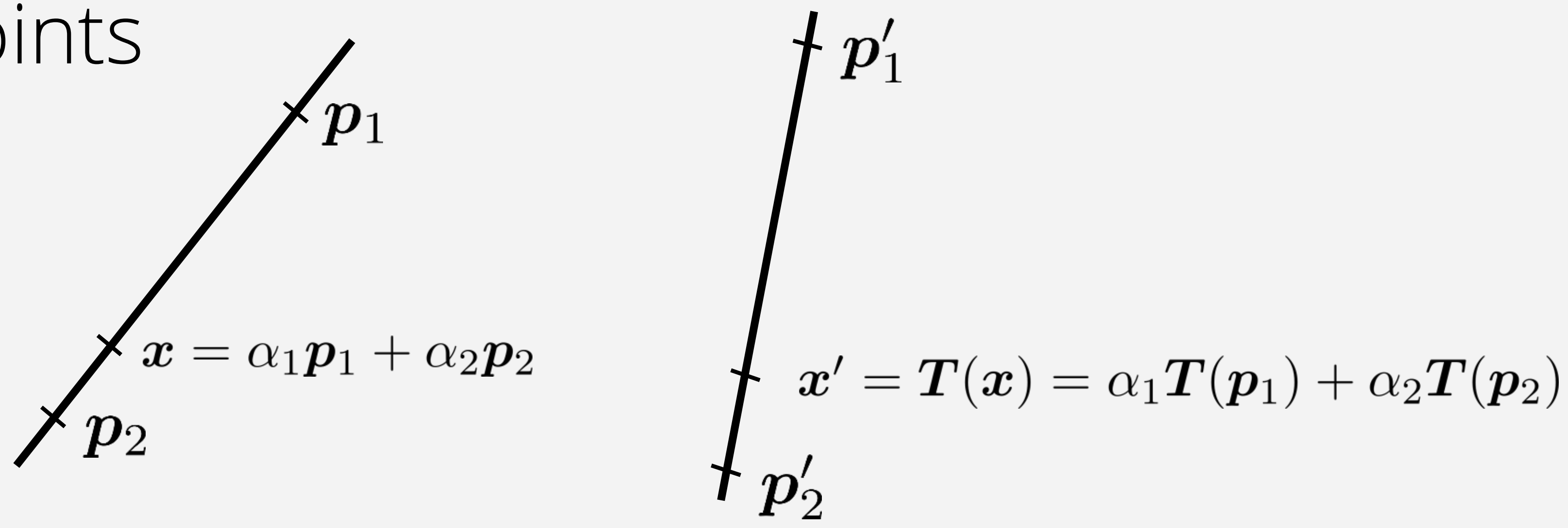
rotation scale etc.

Translation

– 3D position $\boldsymbol{p}$: $\boxed{\boldsymbol{p}' = \boldsymbol{T}(\boldsymbol{p}) = \boldsymbol{A}\boldsymbol{p} + \boldsymbol{t}}$

– Affine transformations preserve affine combinations
$\boldsymbol{T}\left(\sum_i \alpha_i \cdot \boldsymbol{p}_i\right) = \sum_i \alpha_i \cdot \boldsymbol{T}(\boldsymbol{p}_i)$ for $\sum_i \alpha_i = 1$

– E.g., a line can be transformed by transforming its control points



Affine Transformations

- 3D position $\boldsymbol{p}$: $\boldsymbol{p}' = \boldsymbol{A}\boldsymbol{p} + \boldsymbol{t}$
- 3x3 matrix $\boldsymbol{A}$ represents linear transformations
  - Scale, rotation, shear
- 3D vector $\boldsymbol{t}$ represents translation
- Using the homogeneous notation,
  all affine transformations are represented
  with one matrix-vector multiplication

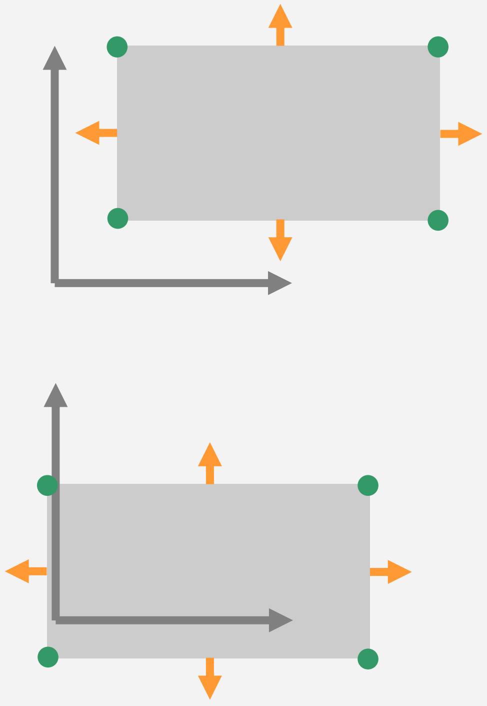
Positions and Vectors

- Positions / vertices specify a location in space
- Vectors / normals specify a direction
- Relations

position	- position	= vector	
position	+ vector	= position	
vector	+ vector	= vector	
position	+ position	not defined	

Positions and Vectors

- Transformations can have different effects on positions and vectors
  - E.g., translation of a point changes its position, but translation of a vector does not change the vector
- Using the homogeneous notation, transformations of vectors and positions are handled in a unified way



Translation of positions and vectors.

Outline

- Motivation
- Homogeneous notation
- Transformations

Homogeneous Coordinates of Positions

homogeneous component

- $[x, y, z, w]^\mathsf{T}$ with $w \neq 0$ are the homogeneous coordinates of the 3D position $(\frac{x}{w}, \frac{y}{w}, \frac{z}{w})^\mathsf{T}$
- $[\lambda x, \lambda y, \lambda z, \lambda w]^\mathsf{T}$ represents the same position $(\frac{\lambda x}{\lambda w}, \frac{\lambda y}{\lambda w}, \frac{\lambda z}{\lambda w})^\mathsf{T} = (\frac{x}{w}, \frac{y}{w}, \frac{z}{w})^\mathsf{T}$ for all $\lambda \neq 0$
- Examples
  - $[2, 3, 4, 1]^\mathsf{T} \sim (2, 3, 4)^\mathsf{T}$
  - $[2, 4, 6, 1]^\mathsf{T} \sim (2, 4, 6)^\mathsf{T}$
  - $[4, 8, 12, 2]^\mathsf{T} \sim (2, 4, 6)^\mathsf{T}$
  - $[0.2, 0.4, 0.6, 0.1]^\mathsf{T} \sim (2, 4, 6)^\mathsf{T}$

Note

$$[x, y, z, w]^\mathsf{T} = \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}$$

Homogeneous Coordinates of Positions

- From Cartesian to homogeneous coordinates

$$(x, y, z)^\mathsf{T} \quad \rightarrow \quad [x, y, z, 1]^\mathsf{T}$$

The most obvious way, but an infinite number of options.

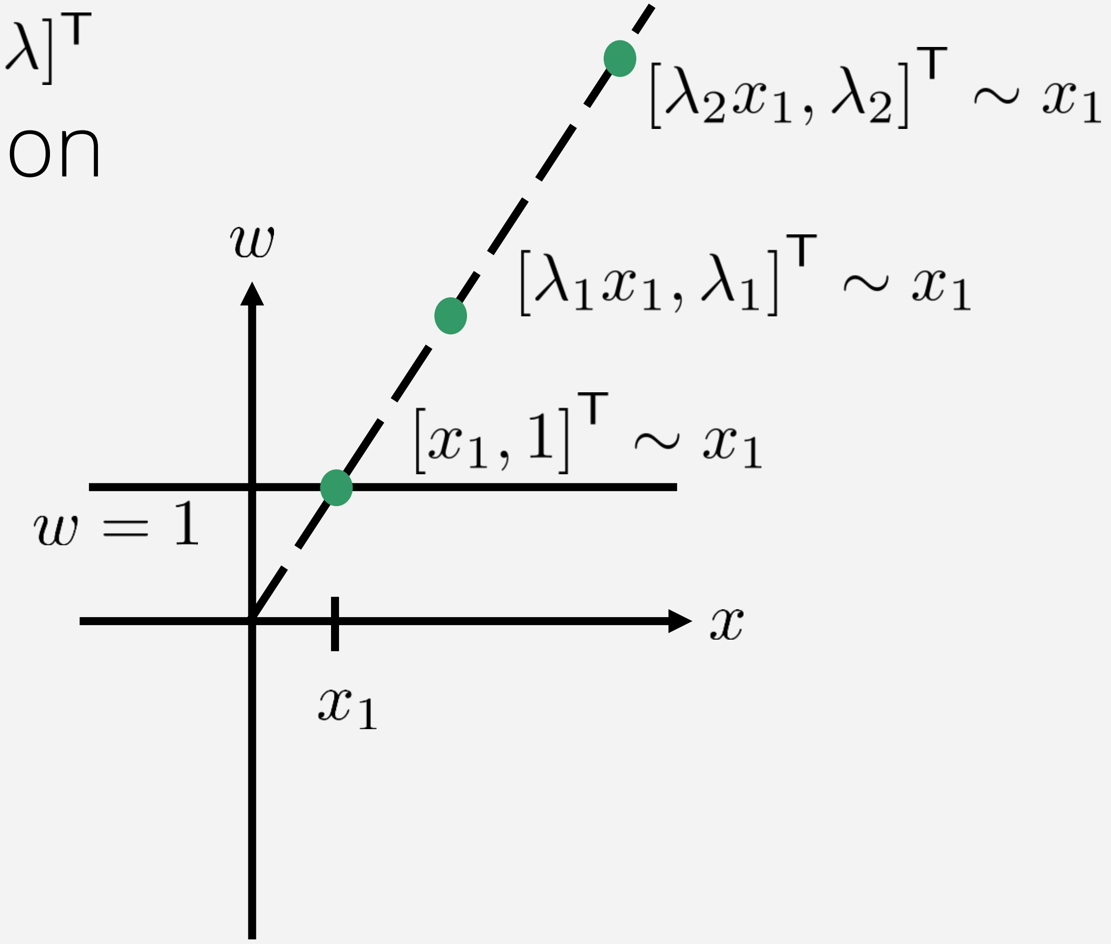
$$(x, y, z)^\mathsf{T} \quad \rightarrow \quad [\lambda x, \lambda y, \lambda z, \lambda]^\mathsf{T} \quad \lambda \neq 0$$

- From homogeneous to Cartesian coordinates

$$[x, y, z, w]^\mathsf{T} \quad \rightarrow \quad (\tfrac{x}{w}, \tfrac{y}{w}, \tfrac{z}{w})^\mathsf{T}$$

1D Illustration

- Homogeneous points $[\lambda x, \lambda]^\mathsf{T}$ represent the same position x in Cartesian space
- Homogeneous points $[\lambda x, \lambda]^\mathsf{T}$ lie on a line in the 2D space $[x, w]$

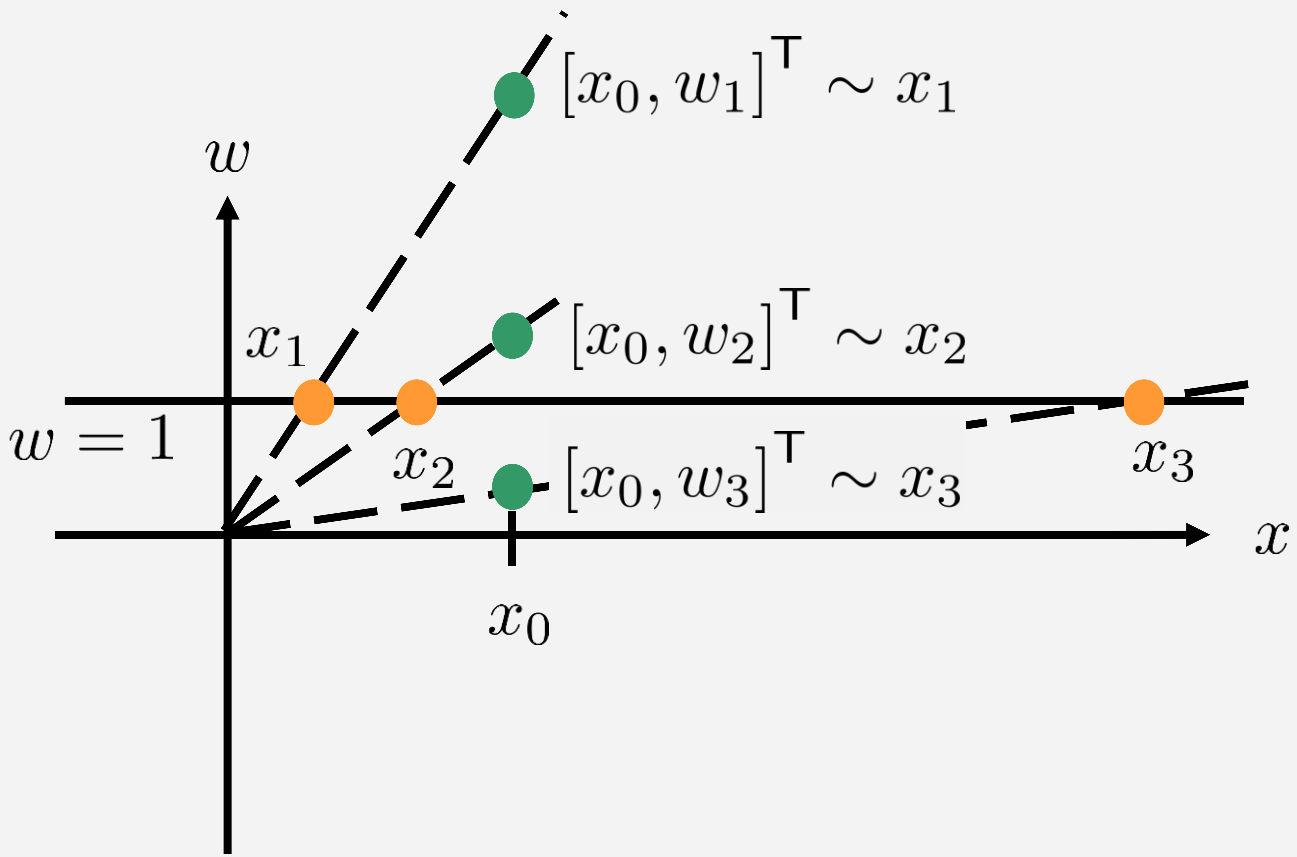


Homogeneous Coordinates of Vectors

- For varying w, a point $[x, y, z, w]^\mathsf{T}$ is scaled and the points $(\frac{x}{w}, \frac{y}{w}, \frac{z}{w})^\mathsf{T}$ represent a line in 3D space
- The direction of this line is $(x, y, z)^\mathsf{T}$
- For $w \to 0$, the position $(\frac{x}{w}, \frac{y}{w}, \frac{z}{w})^\mathsf{T}$ moves to infinity in the direction $(x, y, z)^\mathsf{T}$
- $[x, y, z, 0]^\mathsf{T}$ is a position at infinity in the direction of $(x, y, z)^\mathsf{T}$
- $[x, y, z, 0]^\mathsf{T}$ is a vector in the direction of $(x, y, z)^\mathsf{T}$

UNI FREIBURG

1D Illustration

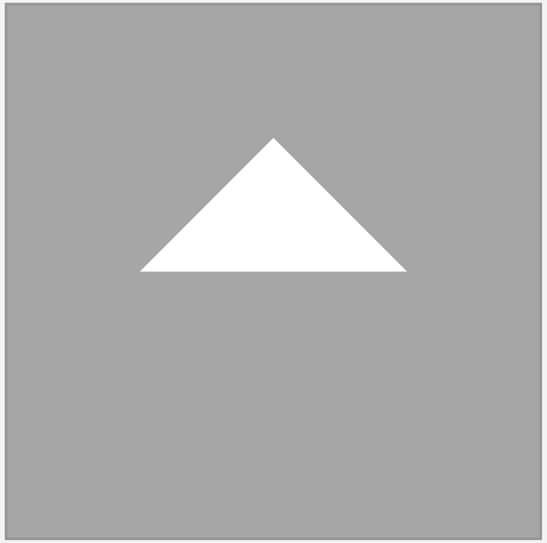


Positions at Infinity

- Can be processed by graphics APIs, e.g. OpenGL
  - Used, e.g. in shadow volumes

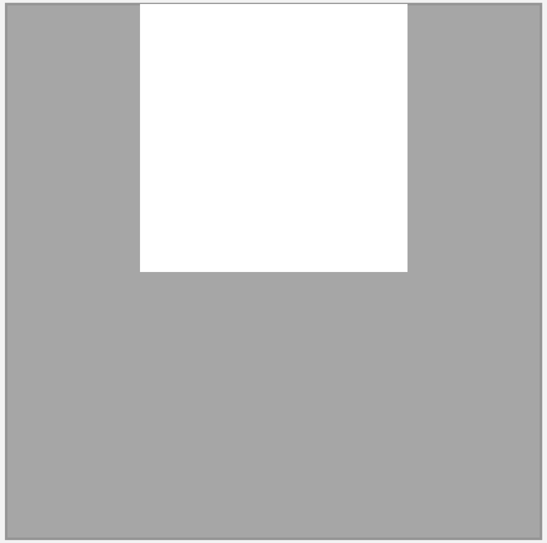
Rendering
of a triangle
with vertices

$$\begin{bmatrix} 0 \\ 0.5 \\ 0 \\ 1 \end{bmatrix} \quad \begin{bmatrix} -0.5 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad \begin{bmatrix} 0.5 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$



Rendering
of a triangle
with vertices

$$\begin{bmatrix} 0 \\ 0.5 \\ 0 \\ 0 \end{bmatrix} \quad \begin{bmatrix} -0.5 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad \begin{bmatrix} 0.5 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$



Positions and Vectors

– If positions are in normalized form,
position-vector relations can be represented

vector + vector = vector

$$\begin{bmatrix} u_x \\ u_y \\ u_z \\ 0 \end{bmatrix} + \begin{bmatrix} v_x \\ v_y \\ v_z \\ 0 \end{bmatrix} = \begin{bmatrix} u_x + v_x \\ u_y + v_y \\ u_z + v_z \\ 0 \end{bmatrix}$$

position + vector = position

$$\begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} + \begin{bmatrix} v_x \\ v_y \\ v_z \\ 0 \end{bmatrix} = \begin{bmatrix} p_x + v_x \\ p_y + v_y \\ p_z + v_z \\ 1 \end{bmatrix}$$

position - position = vector

$$\begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} - \begin{bmatrix} r_x \\ r_y \\ r_z \\ 1 \end{bmatrix} = \begin{bmatrix} p_x - r_x \\ p_y - r_y \\ p_z - r_z \\ 0 \end{bmatrix}$$

Homogeneous Notation of Linear Transformations

$$\begin{pmatrix} m_{00} & m_{01} & m_{02} \\ m_{10} & m_{11} & m_{12} \\ m_{20} & m_{21} & m_{22} \end{pmatrix} \begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix} \sim \begin{bmatrix} m_{00} & m_{01} & m_{02} & 0 \\ m_{10} & m_{11} & m_{12} & 0 \\ m_{20} & m_{21} & m_{22} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix}$$

– If the transform of $\begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix}$ results in $\begin{pmatrix} r_x \\ r_y \\ r_z \end{pmatrix}$, then

the transform of $\begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix}$ results in $\begin{bmatrix} r_x \\ r_y \\ r_z \\ 1 \end{bmatrix} \sim \begin{pmatrix} r_x \\ r_y \\ r_z \end{pmatrix}$

Affine Transformations and Projections

- General form

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

- m_{ij} represent rotation, scale, shear
- t_i represent translation
- p_i are used for projections (see lecture on projections)
- w is the homogeneous component

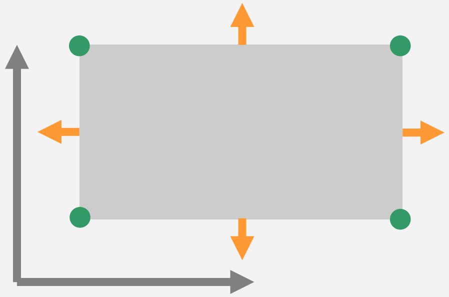
Homogeneous Coordinates - Summary

- $[x, y, z, w]^\mathsf{T}$ 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

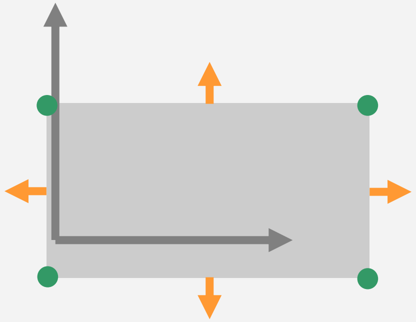
Outline

- Motivation
- Homogeneous notation
- Transformations

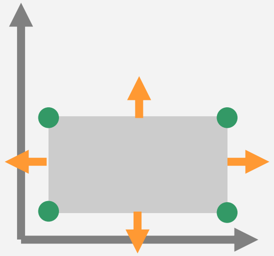
Transformations



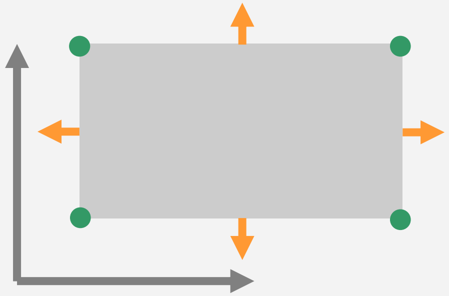
Four faces / primitives / polygons, four points / vertices, four normals.



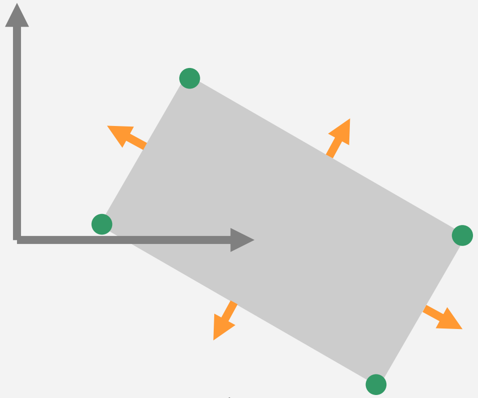
Translation



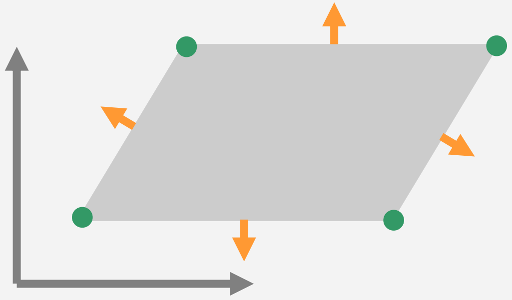
Scale



Identity transform.



Rotation



Shear

Translation

– Of a position

$$\boldsymbol{T}(\boldsymbol{t})\boldsymbol{p} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} = \begin{bmatrix} p_x + t_x \\ p_y + t_y \\ p_z + t_z \\ 1 \end{bmatrix}$$

– Of a vector

$$\boldsymbol{T}(\boldsymbol{t})\boldsymbol{v} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \\ 0 \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ 0 \end{bmatrix}$$

– Inverse transform

$$\boldsymbol{T}^{-1}(\boldsymbol{t}) = \boldsymbol{T}(-\boldsymbol{t})$$

Rotation

- Positive (anticlockwise) rotation with angle ϕ around the *x*-, *y*-, *z*-axis

$$\boldsymbol{R}_x(\phi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi & 0 \\ 0 & \sin\phi & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{R}_y(\phi) = \begin{bmatrix} \cos\phi & 0 & \sin\phi & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\phi & 0 & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{R}_z(\phi) = \begin{bmatrix} \cos\phi & -\sin\phi & 0 & 0 \\ \sin\phi & \cos\phi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Matrices for rotations around arbitrary axes are built by combining simple rotations and translations.

Rotation – Inverse Transform

– The inverse of a rotation matrix is its transpose

$$\boldsymbol{R}_x(-\phi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos -\phi & -\sin -\phi & 0 \\ 0 & \sin -\phi & \cos -\phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \phi & \sin \phi & 0 \\ 0 & -\sin \phi & \cos \phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \boldsymbol{R}_x^{\mathsf{T}}(\phi)$$

$$\boldsymbol{R}_x^{-1} = \boldsymbol{R}_x^{\mathsf{T}} \qquad \boldsymbol{R}_y^{-1} = \boldsymbol{R}_y^{\mathsf{T}} \qquad \boldsymbol{R}_z^{-1} = \boldsymbol{R}_z^{\mathsf{T}}$$

Mirroring / Reflection

- Mirroring with respect to $x = 0$, $y = 0$, $z = 0$ plane
- Changes the sign of the *x*-, *y*-, *z*-component

$$\boldsymbol{P}_x = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \boldsymbol{P}_y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \boldsymbol{P}_z = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- The inverse of a reflection is its transpose

$$\boldsymbol{P}_x^{-1} = \boldsymbol{P}_x^{\mathsf{T}} \qquad \boldsymbol{P}_y^{-1} = \boldsymbol{P}_y^{\mathsf{T}} \qquad \boldsymbol{P}_z^{-1} = \boldsymbol{P}_z^{\mathsf{T}}$$

UNI FREIBURG

Orthogonal Matrices

- Rotation and reflection matrices are orthogonal

  $\boldsymbol{R}\boldsymbol{R}^{\mathsf{T}} = \boldsymbol{R}^{\mathsf{T}}\boldsymbol{R} = \boldsymbol{I} \qquad \boldsymbol{R}^{-1} = \boldsymbol{R}^{\mathsf{T}}$
- $\boldsymbol{R}_1, \boldsymbol{R}_2$ are orthogonal $\Rightarrow$ $\boldsymbol{R}_1\boldsymbol{R}_2$ is orthogonal
- Rotation: $\det \boldsymbol{R} = 1$, Reflection: $\det \boldsymbol{R} = -1$
- Length of a vector is preserved $\|\boldsymbol{R}\boldsymbol{v}\| = \|\boldsymbol{v}\|$
- Angles are preserved $\langle \boldsymbol{R}\boldsymbol{u}, \boldsymbol{R}\boldsymbol{v} \rangle = \langle \boldsymbol{u}, \boldsymbol{v} \rangle$

UNI FREIBURG

Scale

– Scaling *x*-, *y*- , *z*-components of a position or vector

$$\boldsymbol{S}(s_x, s_y, s_z)\boldsymbol{p} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} = \begin{bmatrix} s_x p_x \\ s_y p_y \\ s_z p_z \\ 1 \end{bmatrix}$$

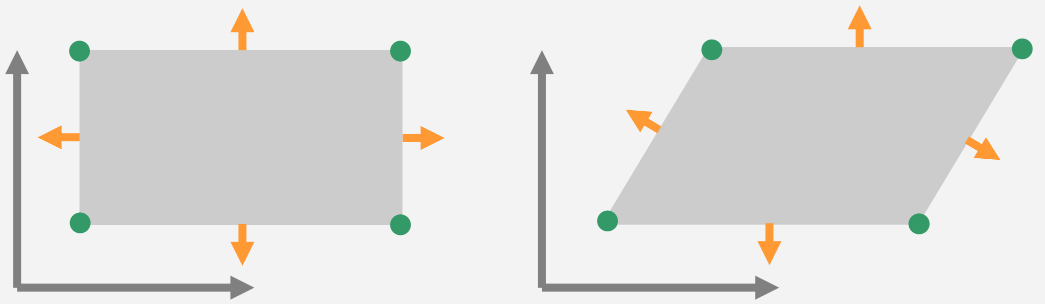
– Inverse $\boldsymbol{S}^{-1}(s_x, s_y, s_z) = \boldsymbol{S}(\frac{1}{s_x}, \frac{1}{s_y}, \frac{1}{s_z})$

– Uniform scaling: $s_x = s_y = s_z = s$

$$\boldsymbol{S}(s) = \begin{bmatrix} s & 0 & 0 & 0 \\ 0 & s & 0 & 0 \\ 0 & 0 & s & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$ or, e.g. $$\boldsymbol{S}(s) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \frac{1}{s} \end{bmatrix}$$

Shear

- Offset of one component with respect to another component
- Six shear modes in 3D
- E.g., shear of *x* with respect to *z*



$$\boldsymbol{H}_{xz}(s)\boldsymbol{p} = \begin{bmatrix} 1 & 0 & s & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} = \begin{bmatrix} p_x + sp_z \\ p_y \\ p_z \\ 1 \end{bmatrix}$$

- Inverse $\boldsymbol{H}_{xz}^{-1}(s) = \boldsymbol{H}_{xz}(-s)$

Compositing Transformations

- Composition is achieved by matrix multiplication
  - A translation $\boldsymbol{T}$ applied to $\boldsymbol{p}$, followed by a rotation $\boldsymbol{R}$

    $\boldsymbol{R}(\boldsymbol{T}\boldsymbol{p}) = (\boldsymbol{R}\boldsymbol{T})\boldsymbol{p}$
  - A rotation $\boldsymbol{R}$ applied to $\boldsymbol{p}$, followed by a translation $\boldsymbol{T}$

    $\boldsymbol{T}(\boldsymbol{R}\boldsymbol{p}) = (\boldsymbol{T}\boldsymbol{R})\boldsymbol{p}$
  - Note that generally $\boldsymbol{T}\boldsymbol{R} \neq \boldsymbol{R}\boldsymbol{T}$
  - The order of composed transformations matters

Examples

– Rotation around a line through $\boldsymbol{t}$ parallel to the *x*-, *y*-, *z*- axis

$$\boldsymbol{T}(\boldsymbol{t})\boldsymbol{R}_{xyz}(\phi)\boldsymbol{T}(-\boldsymbol{t})$$

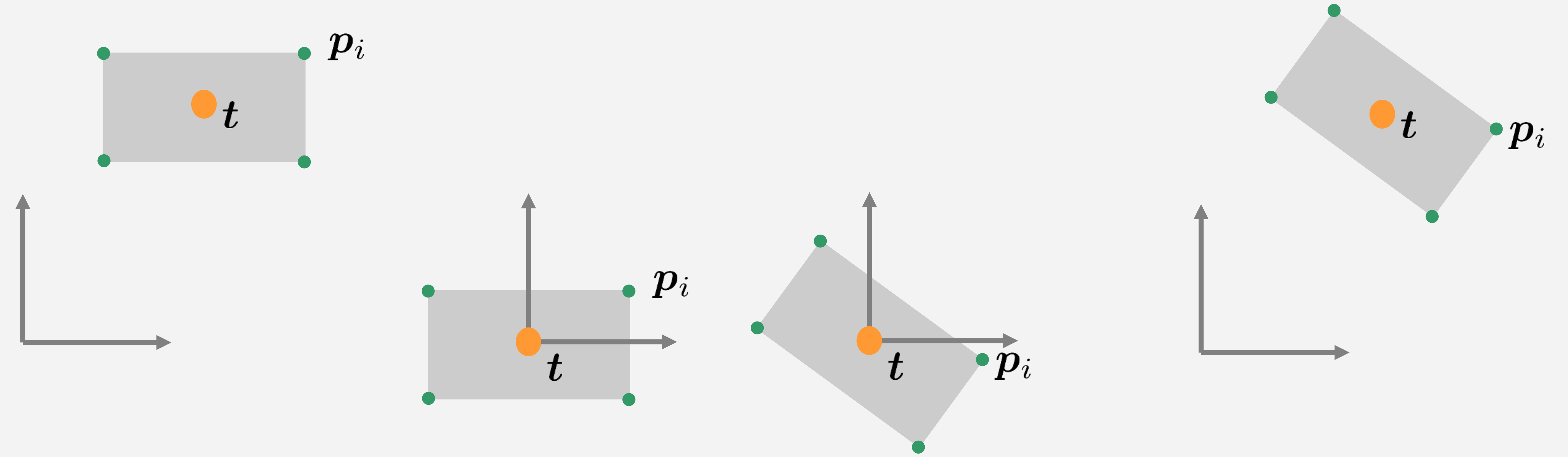
– Scale with respect to an arbitrary axis

$$\boldsymbol{R}_{xyz}(\phi)\boldsymbol{S}(s_x, s_y, s_z)\boldsymbol{R}_{xyz}(-\phi)$$

– E.g., $\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3$ represent an orthonormal basis, then scaling along these vectors is realized with

$$\begin{bmatrix} \boldsymbol{b}_1 & \boldsymbol{b}_2 & \boldsymbol{b}_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \boldsymbol{S}(s_x, s_y, s_z) \begin{bmatrix} \boldsymbol{b}_1 & \boldsymbol{b}_2 & \boldsymbol{b}_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{\mathsf{T}}$$

2D Example – Rotation About a Point



We want to rotate the object points p_i around point t.

Center in the origin.

Translation by $-t$.

$$\boldsymbol{T}(-\boldsymbol{t})\boldsymbol{p}_i$$

Rotation by ϕ.

$$\boldsymbol{R}(\phi)\boldsymbol{T}(-\boldsymbol{t})\boldsymbol{p}_i$$

Translation by t.

$$\boldsymbol{T}(\boldsymbol{t})\boldsymbol{R}(\phi)\boldsymbol{T}(-\boldsymbol{t})\boldsymbol{p}_i$$

Rigid-Body Transform

- In Cartesian coordinates: $\boldsymbol{p}' = \boldsymbol{R}\boldsymbol{p} + \boldsymbol{t}$ with $\boldsymbol{R}$ being a rotation and $\boldsymbol{t}$ being a translation
- In homogeneous notation: $\begin{bmatrix} \boldsymbol{p}' \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0}^\top & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{p} \\ 1 \end{bmatrix}$
- The inverse transform in Cartesian coordinates

$$\boldsymbol{p} = \boldsymbol{R}^{-1}(\boldsymbol{p}' - \boldsymbol{t}) = \boldsymbol{R}^{-1}\boldsymbol{p}' - \boldsymbol{R}^{-1}\boldsymbol{t} = \boldsymbol{R}^\top\boldsymbol{p}' - \boldsymbol{R}^\top\boldsymbol{t}$$

- The inverse in homogeneous notation

$$\begin{bmatrix} \boldsymbol{p} \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0}^\top & 1 \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{p}' \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}^\top & -\boldsymbol{R}^\top\boldsymbol{t} \\ \boldsymbol{0}^\top & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{p}' \\ 1 \end{bmatrix}$$

Planes and Normals

– Planes can be represented by a surface normal $\boldsymbol{n}$ and a point $\boldsymbol{r}$. All points $\boldsymbol{p}$ with $\boldsymbol{n} \cdot (\boldsymbol{p} - \boldsymbol{r}) = 0$ form a plane

$$n_x p_x + n_y p_y + n_z p_z + (-n_x r_x - n_y r_y - n_z r_z) = 0$$

$$n_x p_x + n_y p_y + n_z p_z + d = 0$$

$$(n_x \; n_y \; n_z \; d)(p_x \; p_y \; p_z \; 1)^\mathsf{T} = 0$$

$$(n_x \; n_y \; n_z \; d)\boldsymbol{A}^{-1}\boldsymbol{A}(p_x \; p_y \; p_z \; 1)^\mathsf{T} = 0$$

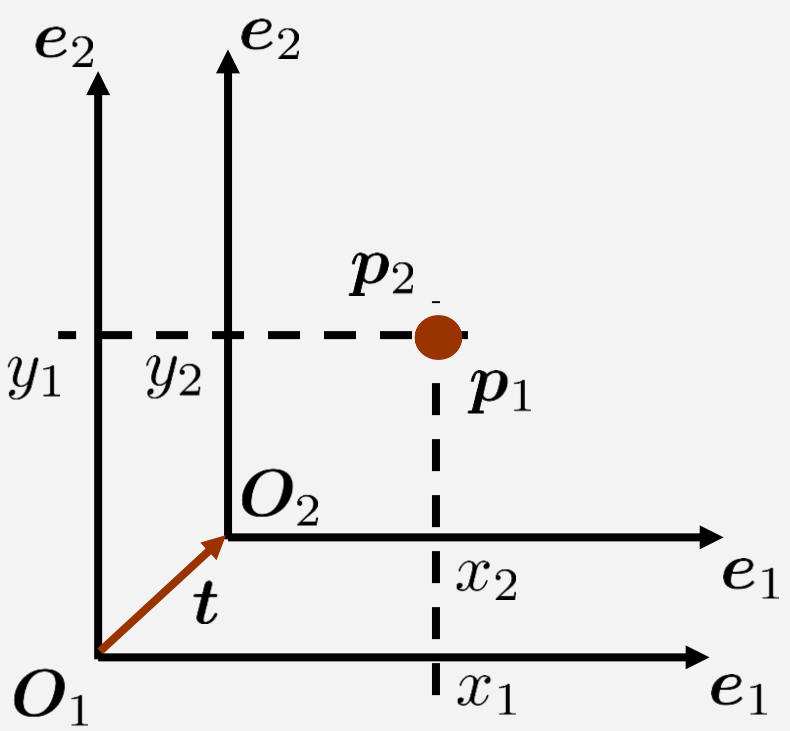
– The transformed points $\boldsymbol{A}[p_x \; p_y \; p_z \; 1]^\mathsf{T}$ are on the plane represented by $(n_x \; n_y \; n_z \; d)\boldsymbol{A}^{-1} = ((\boldsymbol{A}^{-1})^\mathsf{T}(n_x \; n_y \; n_z \; d)^\mathsf{T})^\mathsf{T}$

– If a surface is transformed by $\boldsymbol{A}$, its homogeneous notation (including the normal) is transformed by $(\boldsymbol{A}^{-1})^\mathsf{T}$

Basis Transform - Translation

– Two coordinate systems

$$\boldsymbol{C}_1 = (\boldsymbol{O}_1, \{\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3\})$$
$$\boldsymbol{C}_2 = (\boldsymbol{O}_2, \{\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3\})$$
$$\boldsymbol{O}_2 = \boldsymbol{T}(\boldsymbol{t})\boldsymbol{O}_1$$



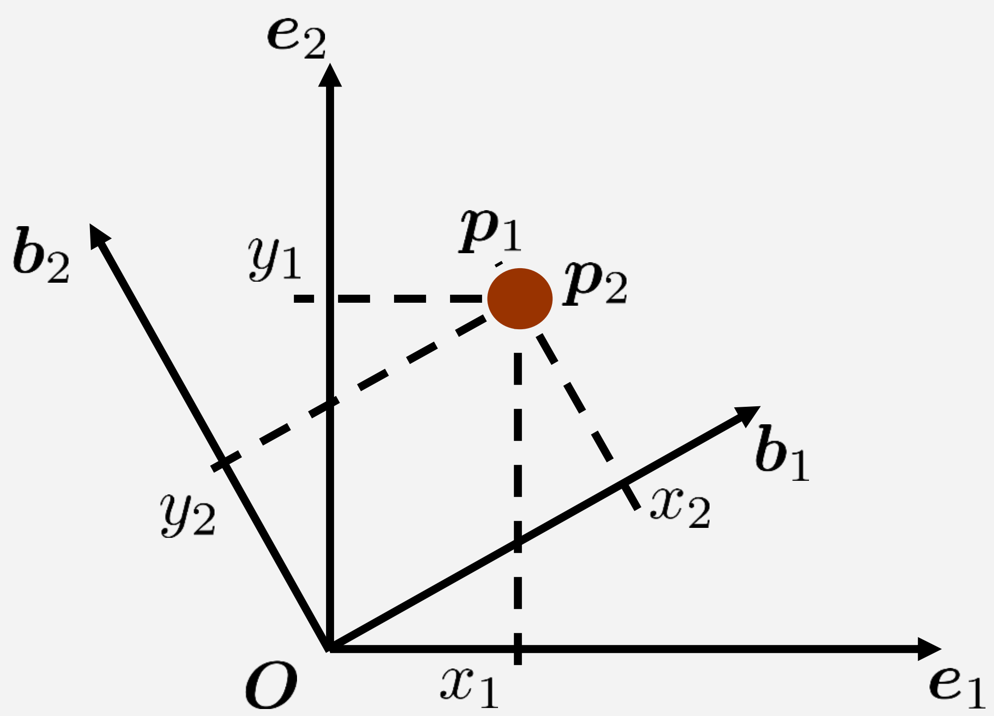
Basis Transform - Translation

- The coordinates of $\boldsymbol{p}_1$ with respect to $\boldsymbol{C}_2$ are given by $\boldsymbol{p}_2 = \boldsymbol{p}_1 - \boldsymbol{t}$ $\quad \boldsymbol{p}_2 = \boldsymbol{T}(-\boldsymbol{t})\boldsymbol{p}_1$
- The coordinates of a point in the transformed basis correspond to the coordinates of the point in the untransformed basis transformed by the inverse basis transform
  - Translating the origin by $\boldsymbol{t}$ corresponds to translating the object by $-\boldsymbol{t}$
  - Rotating the basis vectors by an angle corresponds to rotating the object by the same negative angle

Basis Transform - Rotation

– Two coordinate systems

$$C_1 = (\boldsymbol{O}, \{\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3\})$$
$$C_2 = (\boldsymbol{O}, \{\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3\})$$



Basis Transform - Rotation

- Coordinates of $\boldsymbol{p}_1$ with respect to $\boldsymbol{C}_2$ are given by

$$\boldsymbol{p}_2 = \begin{pmatrix} \boldsymbol{b}_1^\top \\ \boldsymbol{b}_2^\top \\ \boldsymbol{b}_3^\top \end{pmatrix} \boldsymbol{p}_1 \sim \begin{bmatrix} \boldsymbol{b}_{1,x} & \boldsymbol{b}_{1,y} & \boldsymbol{b}_{1,z} & 0 \\ \boldsymbol{b}_{2,x} & \boldsymbol{b}_{2,y} & \boldsymbol{b}_{2,z} & 0 \\ \boldsymbol{b}_{3,x} & \boldsymbol{b}_{3,y} & \boldsymbol{b}_{3,z} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \boldsymbol{p}_1$$

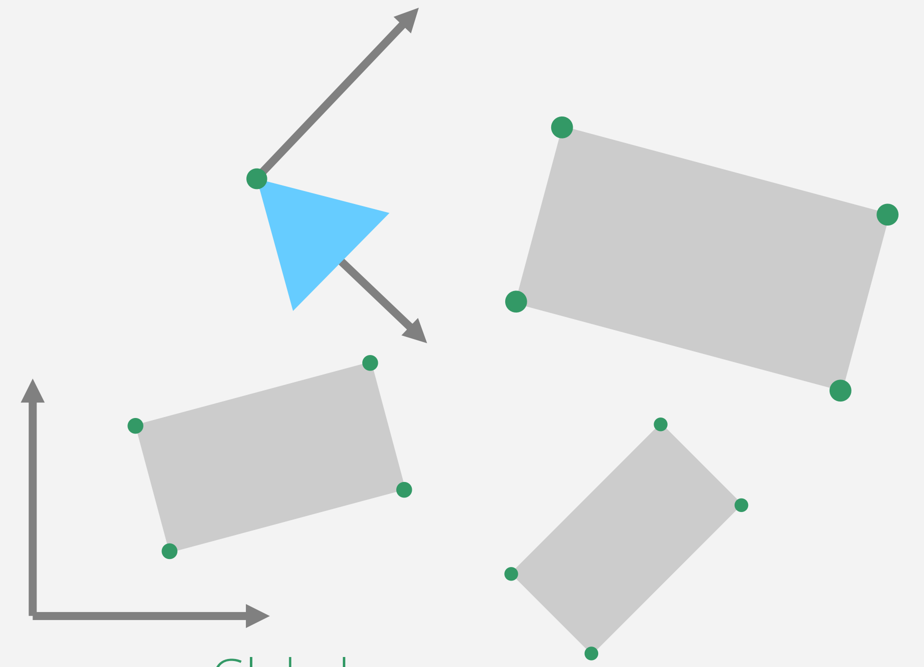
- $\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3$ are the basis vectors of $\boldsymbol{C}_2$ with respect to $\boldsymbol{C}_1$
- $\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3$ are orthonormal, represent a rotation
- Rotating the basis vectors by an angle corresponds to rotating the object by the same negative angle

Basis Transform - Application

- The view transform can be seen as a basis transform
- Objects are in a global system $\boldsymbol{C}_1 = (\boldsymbol{O}_1, \{\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3\})$
- The camera is at $\boldsymbol{O}_2$ and oriented with $\{\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3\}$
- After the view transform, all objects are represented in the eye or camera coordinate system $\boldsymbol{C}_2 = (\boldsymbol{O}_2, \{\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3\})$
- Placing and orienting the camera is a transformation $\boldsymbol{V}$
- The basis transform is realized by applying $\boldsymbol{V}^{-1}$ to all objects

View Transform

$$\boldsymbol{C}_2 = (\boldsymbol{O}_2, \{\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3\})$$

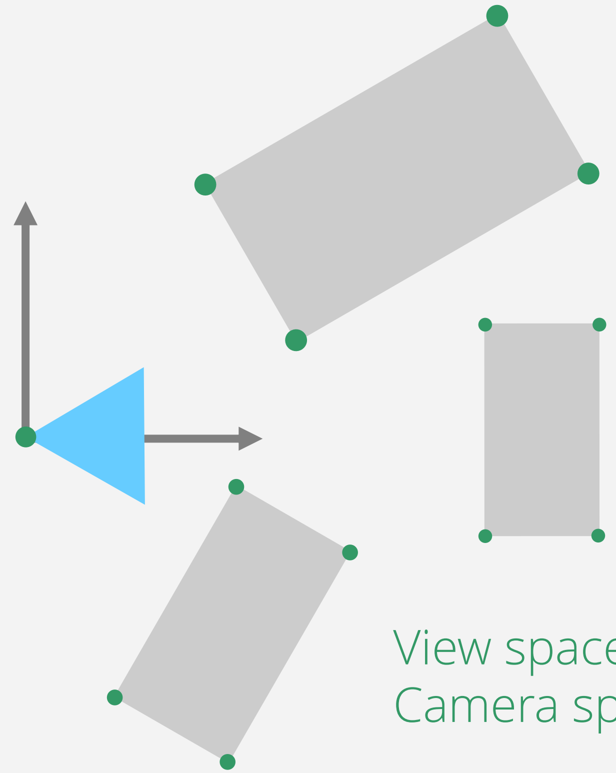


Global space.

$$\boldsymbol{C}_1 = (\boldsymbol{O}_1, \{\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3\})$$

$$\boldsymbol{C}_1 \quad \rightarrow \quad \boldsymbol{V} \quad \rightarrow \quad \boldsymbol{C}_2$$

Inverse view transform V^{-1} applied to all objects and the camera



View space / Camera space.

Summary

- Usage of the homogeneous notation is motivated by a unified processing of affine transformations, perspective projections, points, and vectors
- All transformations of points and vectors are represented by a matrix-vector multiplication
- “Undoing" a transformation is represented by its inverse
- Compositing of transformations is accomplished by matrix multiplication