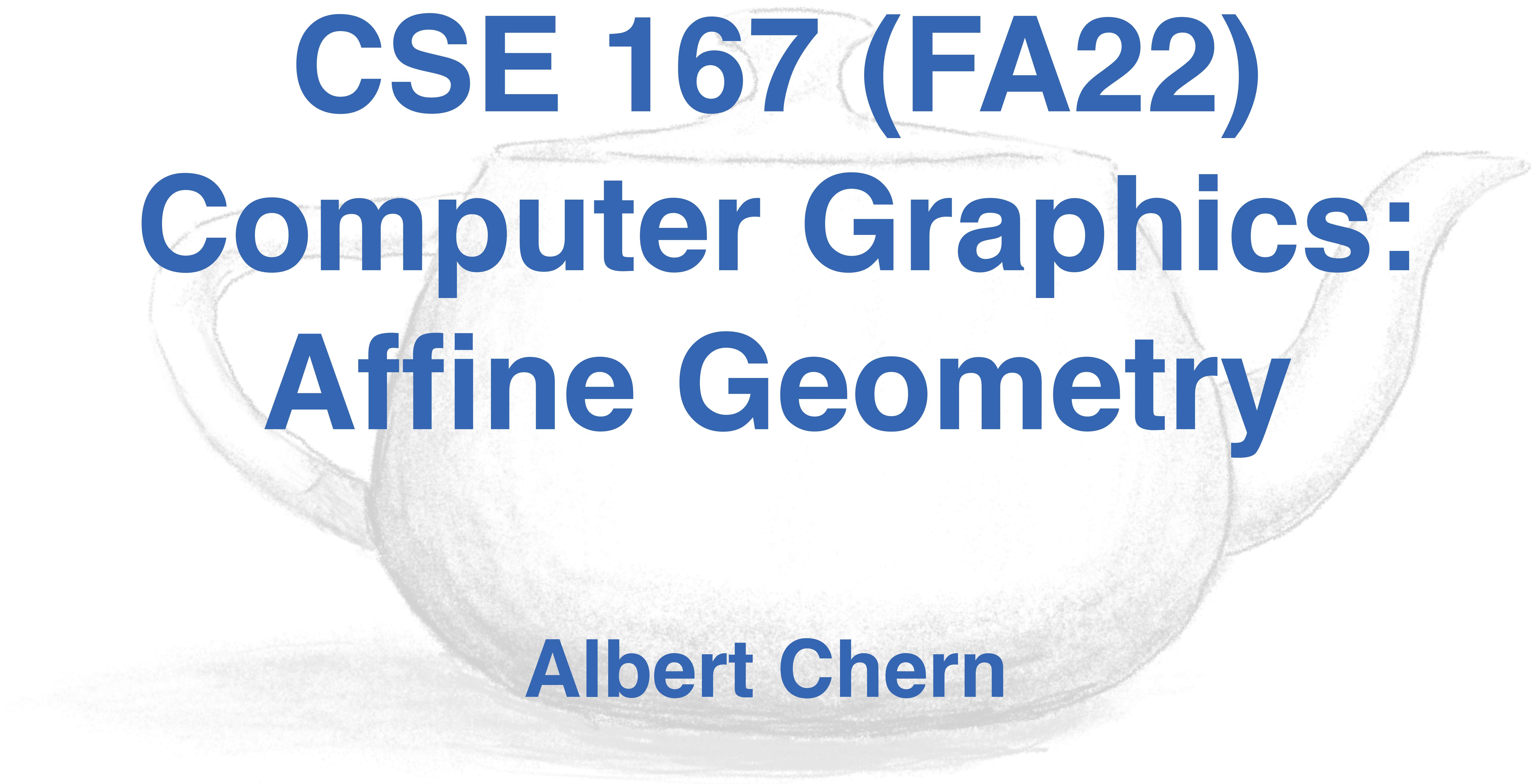
CSE 167 (FA22)
Computer Graphics:
Affine Geometry



Albert Chern

UC San Diego
JACOBS SCHOOL OF ENGINEERING

Overview

- We've learned about the geometry and algebra of *vectors*.
- Today: *positions (affine points)* *and* *displacements (vectors)*
- Coordinate system (analogous to basis)
- Affine transformations = linear transforms + translations
- model matrix, view matrix

Recall: Vectors

- Recall: vectors
- Affine points
- Coordinate systems
- Affine transformations
- Model/camera/view
- View matrix

Recall: Vectors

- There are two representations of vectors:

Geometric $\vec{v} \in V$ — vector space

Algebraic $\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \in \mathbb{R}^3$

- A **basis** of the vector space relates the two representations

basis

$$\vec{v} = \begin{bmatrix} \vec{e}_1 & \vec{e}_2 & \vec{e}_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

- Transformations and change of basis

$$\begin{bmatrix} \vec{a}_1 & \vec{a}_2 & \vec{a}_3 \end{bmatrix} = \begin{bmatrix} \vec{e}_1 & \vec{e}_2 & \vec{e}_3 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

coefficients for $\vec{a}_2$ under basis $(\vec{e}_1, \vec{e}_2, \vec{e}_3)$

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

$$\mathbb{R}^3_{\vec{\mathbf{e}}} \longleftarrow \mathbb{R}^3_{\vec{\mathbf{a}}}$$

$$\mathbf{Av} \qquad \mathbf{v}$$

- Basic operations
  - Vector add vector $\vec{u} + \vec{v}$ $\qquad \mathbf{u} + \mathbf{v}$
  - scalar times vector $a\vec{v}$ $\qquad a\mathbf{v}$

Limitations of vectors

- Vectors are good at describing
  - Displacements, velocities, accelerations, forces
- Vectors are awkward at describing point positions
  - Additions and scalings are meaningless for point positions
  - Linear transformation cannot represent translations (parallel shift)
- Next, we will mix in a new type of object called (affine) **point**

Affine points

- Recall: vectors
- Affine points
- Coordinate systems
- Affine transformations
- Model/camera/view
- View matrix

Positions and Displacements

Points and **vectors** are different.

- **Points** describe **positions** (denoted by $\underline{p}$)
- **Vectors** describe **displacements** (denoted by $\vec{v}$)

Base on our experience:

- Differences of positions are displacements $\underline{q} - \underline{p} = \vec{v}$
- Position add displacement is position $\underline{p} + \vec{v} = \underline{q}$
- Linear combinations of displacements are displacements

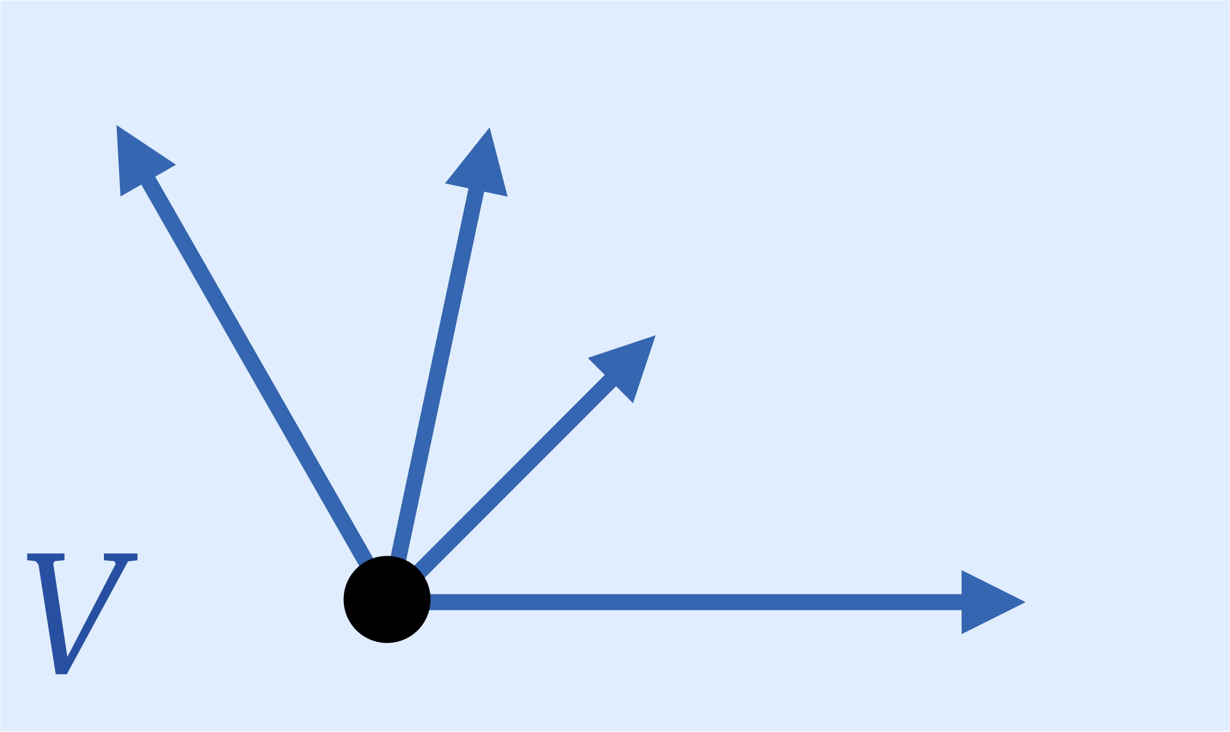
$$c_1\vec{v}_1 + c_2\vec{v}_2 = \vec{u}$$

- Scale and sum of positions don't make sense $c_1\underline{p}_1 + c_2\underline{p}_2 = ?$
- Average of positions is fine $0.5\underline{p}_1 + 0.5\underline{p}_2 = \underline{q}$

Positions and Displacements

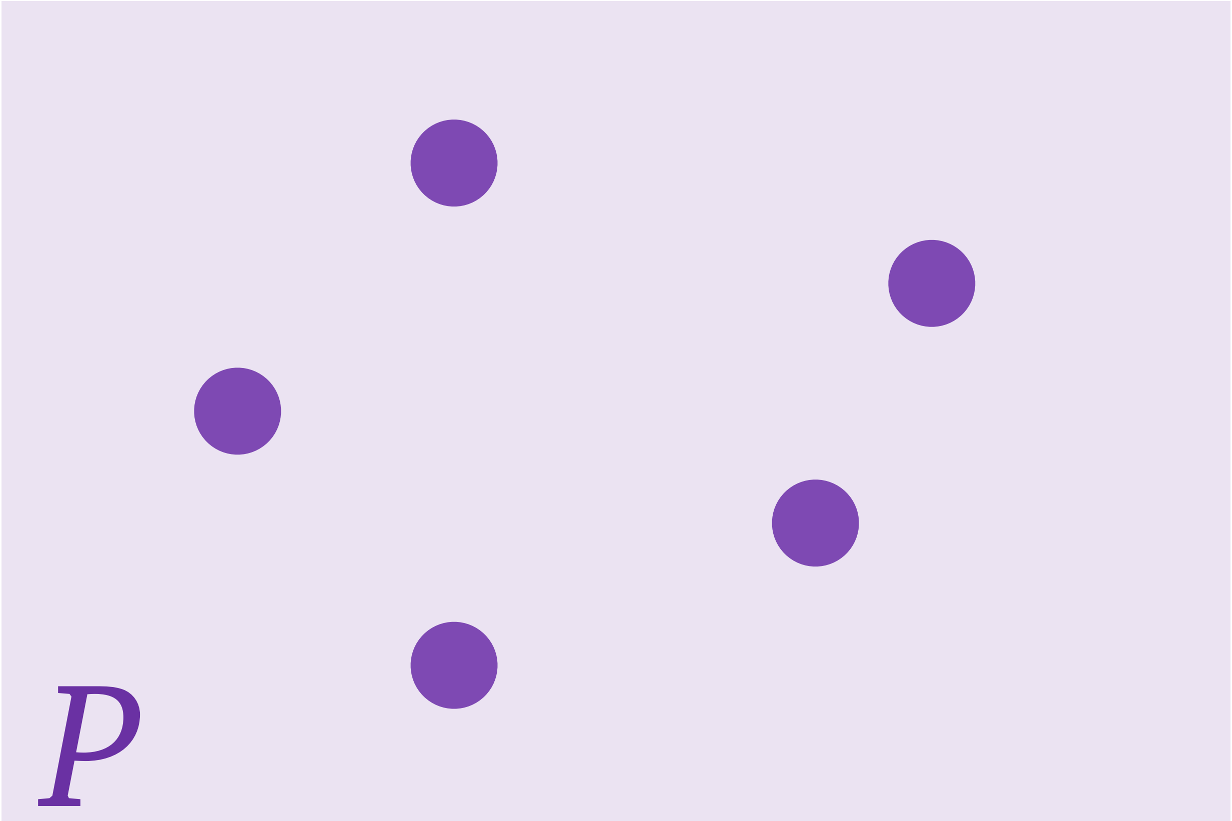
The collection of all **vectors** form a **vector space**.

- $\overrightarrow{\text{vec}} + \overrightarrow{\text{vec}} = \overrightarrow{\text{vec}}$
- scalar * $\overrightarrow{\text{vec}} = \overrightarrow{\text{vec}}$



The collection of all **points** form an **affine space**.

- $\underline{\text{pt}} + \overrightarrow{\text{vec}} = \underline{\text{pt}}$
- $\underline{\text{pt}} - \underline{\text{pt}} = \overrightarrow{\text{vec}}$
- scalar * $\underline{\text{pt}}$ = not defined!
- $\underline{\text{pt}} + \underline{\text{pt}}$ = not defined!



Positions and Displacements

What is the **algebraic representation** for vectors and points?

Homogeneous coordinates

Positions and Displacements

Points/positions

$$\begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix}$$

Vectors/displacements

$$\begin{bmatrix} v_x \\ v_y \\ v_z \\ 0 \end{bmatrix}$$

Positions and Displacements

Points/positions

$$\begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{q} \\ 1 \end{bmatrix} - \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{q} - \mathbf{p} \\ 0 \end{bmatrix}$$

pt pt vec

Vectors/displacements

$$\begin{bmatrix} v_x \\ v_y \\ v_z \\ 0 \end{bmatrix}$$

$$c_1 \begin{bmatrix} \mathbf{v}_1 \\ 0 \end{bmatrix} + c_2 \begin{bmatrix} \mathbf{v}_2 \\ 0 \end{bmatrix} = \begin{bmatrix} c_1\mathbf{v}_1 + c_2\mathbf{v}_2 \\ 0 \end{bmatrix}$$

vec vec vec

Positions and Displacements

$$\begin{bmatrix}\mathbf{q}\\1\end{bmatrix}-\begin{bmatrix}\mathbf{p}\\1\end{bmatrix}=\begin{bmatrix}\mathbf{q}-\mathbf{p}\\0\end{bmatrix}$$

pt pt vec

$$c_1\begin{bmatrix}\mathbf{v}_1\\0\end{bmatrix}+c_2\begin{bmatrix}\mathbf{v}_2\\0\end{bmatrix}=\begin{bmatrix}c_1\mathbf{v}_1+c_2\mathbf{v}_2\\0\end{bmatrix}$$

vec vec vec

$$\begin{bmatrix}\mathbf{p}_1\\1\end{bmatrix}+\begin{bmatrix}\mathbf{p}_2\\1\end{bmatrix}=\begin{bmatrix}\mathbf{p}_1+\mathbf{p}_2\\2\end{bmatrix}$$

pt pt not defined

$$0.5\begin{bmatrix}\mathbf{p}_1\\1\end{bmatrix}+0.5\begin{bmatrix}\mathbf{p}_2\\1\end{bmatrix}=\begin{bmatrix}\frac{\mathbf{p}_1+\mathbf{p}_2}{2}\\1\end{bmatrix}$$

pt pt pt

Positions and Displacements

Points/positions

$$\begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix}$$

Vectors/displacements

$$\begin{bmatrix} v_x \\ v_y \\ v_z \\ 0 \end{bmatrix}$$

The 4-dimensional coordinates for 3D points and vectors are called the **homogeneous coordinates**.

Coordinate system

- Recall: vectors
- Affine points
- **Coordinate systems**
- Affine transformations
- Model/camera/view
- View matrix

Basis v.s. coordinate systems

- For vectors $\vec{v} = \begin{bmatrix} \vec{e}_1 & \vec{e}_2 & \vec{e}_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$

  basis

- For affine points

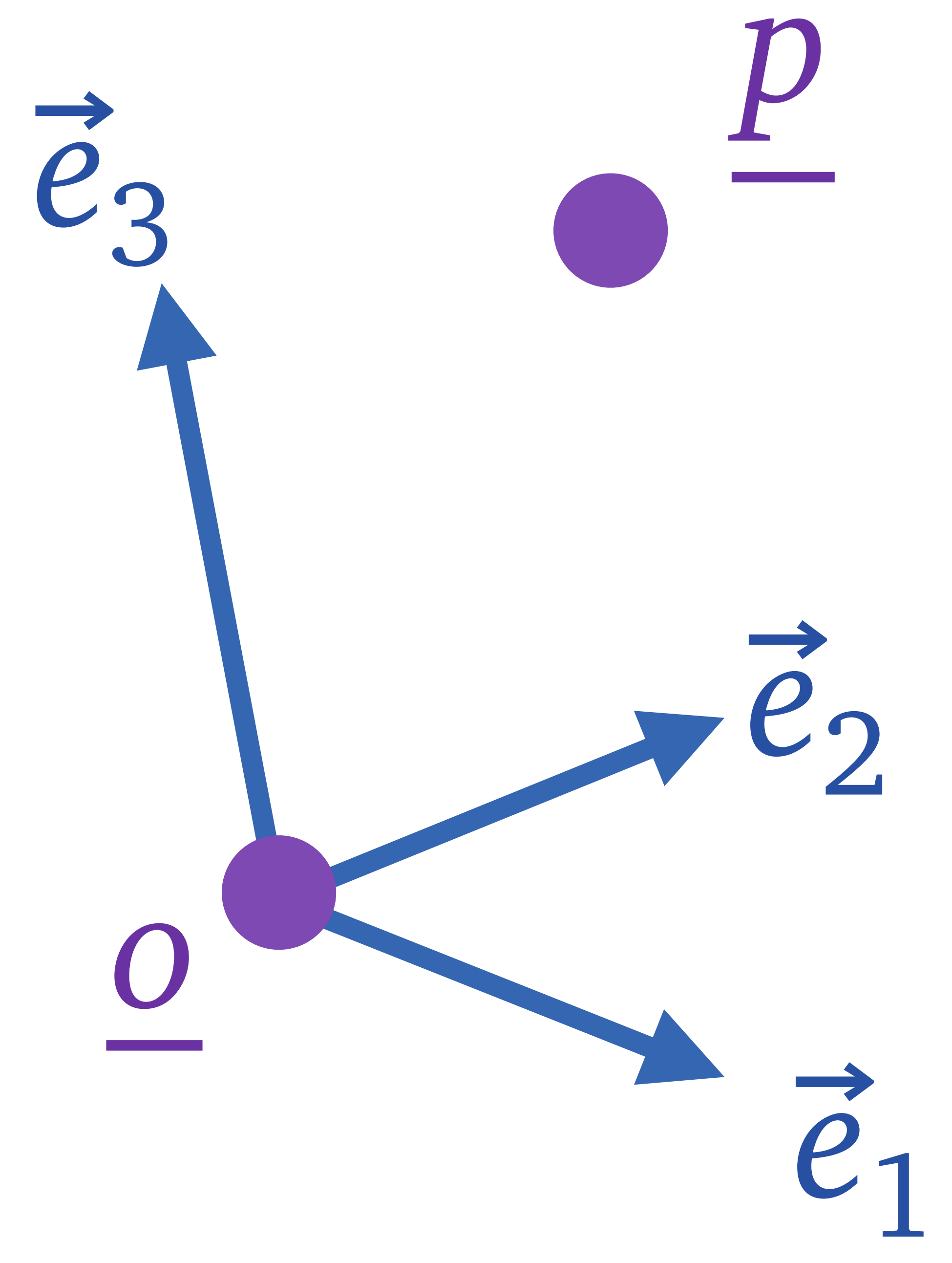
basis *origin*

$$\underline{p} = \begin{bmatrix} \vec{e}_1 & \vec{e}_2 & \vec{e}_3 & \underline{o} \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ 1 \end{bmatrix}$$

coordinate system

$$= p_1\vec{e}_1 + p_2\vec{e}_2 + p_3\vec{e}_3 + \underline{o}$$

displacement vector $\underline{p} - \underline{o}$



Coordinate system

Let P be an affine space modeled on a vector space V.

A **frame** or an **affine coordinate system** is a basis $\vec{\mathbf{e}}$ for V together with a point $\underline{o} \in P$.

Given any $\underline{p} \in P$ we can uniquely write it as

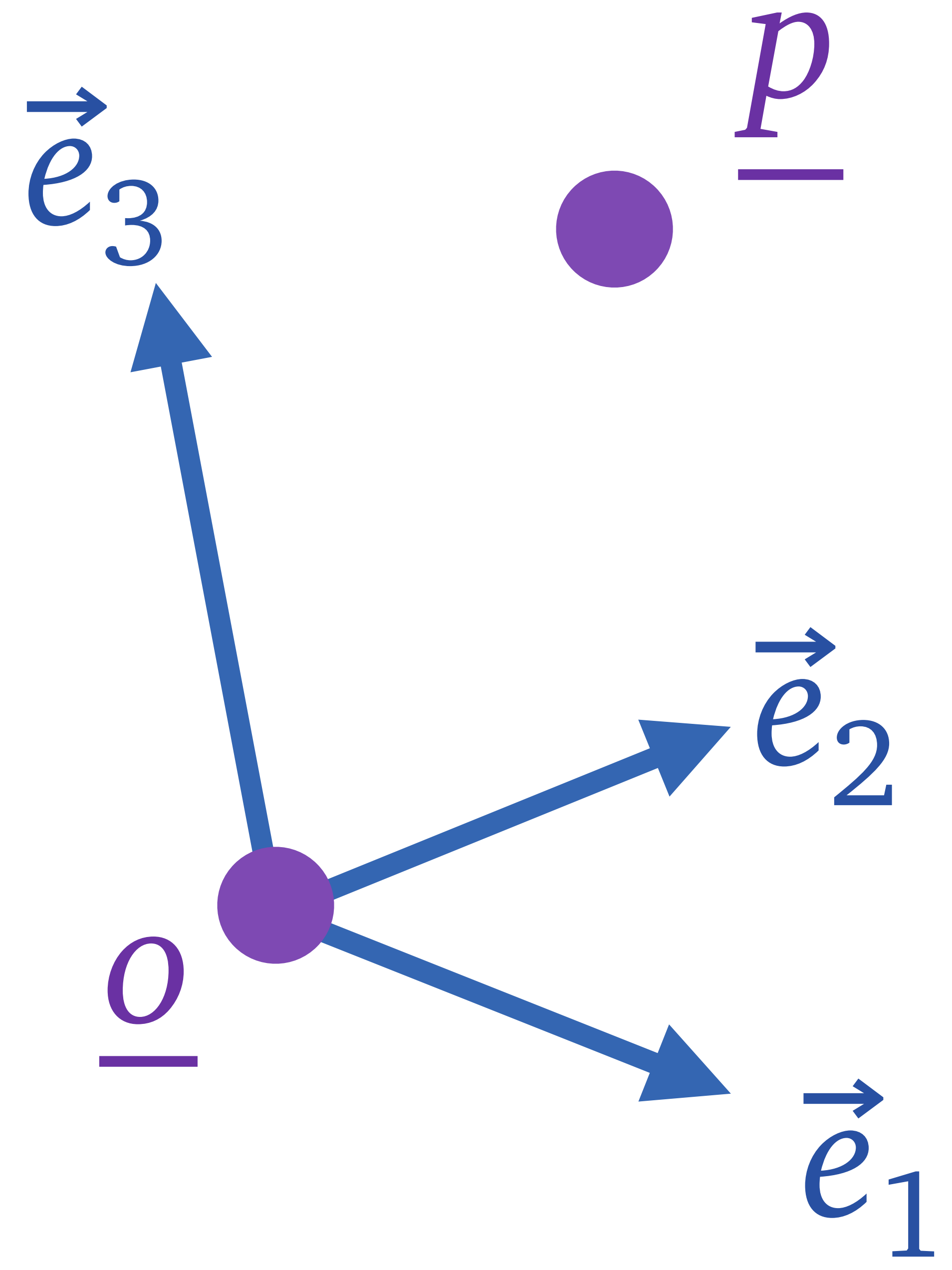
$$\underline{p} = p_1\vec{e}_1 + \cdots + p_n\vec{e}_n + \underline{o}$$

$$= \begin{bmatrix} \vec{\mathbf{e}}^\intercal & \underline{o} \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix}$$

geometric point

coordinate system

array of numbers as homogeneous coordinate



Affine Transformations

- Recall: vectors
- Affine points
- Coordinate systems
- Affine transformations
- Model/camera/view
- View matrix

Affine transformations

- In matrix algebra, we call $\mathbf{x} \mapsto \mathbf{A}\mathbf{x}$ linear transformations

$$\begin{bmatrix} \mathbf{x}_{3\times 1} \end{bmatrix} \mapsto \begin{bmatrix} \mathbf{A}_{3\times 3} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{3\times 1} \end{bmatrix}$$

- Transformations that take the form of $\mathbf{x} \mapsto \mathbf{A}\mathbf{x} + \mathbf{b}$

$$\begin{bmatrix} \mathbf{x}_{3\times 1} \end{bmatrix} \mapsto \begin{bmatrix} \mathbf{A}_{3\times 3} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{3\times 1} \end{bmatrix} + \begin{bmatrix} \mathbf{b}_{3\times 1} \end{bmatrix}$$

are called **affine transformations**.

Affine transformations

- Transformations that take the form of $\mathbf{x} \mapsto \mathbf{A}\mathbf{x} + \mathbf{b}$

$$\begin{bmatrix} \mathbf{x}_{3\times1} \end{bmatrix} \mapsto \begin{bmatrix} \mathbf{A}_{3\times3} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{3\times1} \end{bmatrix} + \begin{bmatrix} \mathbf{b}_{3\times1} \end{bmatrix}$$

are called **affine transformations**.

- Using the homogeneous coordinate, it's a single matrix multiplication!

$$\begin{bmatrix} \mathbf{x}_{3\times1} \\ 1 \end{bmatrix} \mapsto \begin{bmatrix} \mathbf{A}_{3\times3} & \mathbf{b}_{3\times1} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x}_{3\times1} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{A}\mathbf{x} + \mathbf{b} \\ 1 \end{bmatrix}$$

Affine transformation matrix

- An affine transformation matrix takes the form

linear part *translation part*

$$\mathbf{M} = \begin{bmatrix} A_{11} & A_{12} & A_{13} & b_1 \\ A_{21} & A_{22} & A_{23} & b_2 \\ A_{31} & A_{32} & A_{33} & b_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

last row is always 0001

Relating two coordinate systems

- Suppose we have two coordinate systems $\begin{bmatrix} \vec{e}'_1 & \vec{e}'_2 & \vec{e}'_3 & \underline{o}' \end{bmatrix}$ $\begin{bmatrix} \vec{e}_1 & \vec{e}_2 & \vec{e}_3 & \underline{o} \end{bmatrix}$
- They are related by an affine matrix

linear part

translation part

coordinates for $\vec{e}'_2$ under coord. system $(\vec{e}_1, \vec{e}_2, \vec{e}_3, \underline{o})$

$$\begin{bmatrix} \vec{e}'_1 & \vec{e}'_2 & \vec{e}'_3 & \underline{o}' \end{bmatrix} = \begin{bmatrix} \vec{e}_1 & \vec{e}_2 & \vec{e}_3 & \underline{o} \end{bmatrix} \begin{bmatrix} A_{11} & A_{12} & A_{13} & b_1 \\ A_{21} & A_{22} & A_{23} & b_2 \\ A_{31} & A_{32} & A_{33} & b_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

coordinates for $\underline{o}'$ under coord. system $(\vec{e}_1, \vec{e}_2, \vec{e}_3, \underline{o})$

  - The linear part relates the bases $\vec{\mathbf{e}}'^{\mathsf{T}} = \vec{\mathbf{e}}^{\mathsf{T}} \mathbf{A}$
  - The translation part relates the origin $\underline{o}' = \underline{o} + \vec{\mathbf{e}}^{\mathsf{T}} \mathbf{b}$

Relating two coordinate systems

$$\begin{bmatrix} \vec{e}'_1 & \vec{e}'_2 & \vec{e}'_3 & \underline{o}' \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ 1 \end{bmatrix} = \begin{bmatrix} \vec{e}_1 & \vec{e}_2 & \vec{e}_3 & \underline{o} \end{bmatrix} \overbrace{\begin{bmatrix} A_{11} & A_{12} & A_{13} & b_1 \\ A_{21} & A_{22} & A_{23} & b_2 \\ A_{31} & A_{32} & A_{33} & b_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}}^{\mathbf{M}} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ 1 \end{bmatrix}$$

coordinates of $\underline{p}$ under the $(\vec{\mathbf{e}}', \underline{o}')$ coordinate system

$\mathbf{M}\begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix}$ is thecoordinates of $\underline{p}$ under the $(\vec{\mathbf{e}}, \underline{o})$ coordinate system

$$\mathbf{M} = \begin{bmatrix} \mathbf{A}_{3\times3} & \mathbf{b}_{3\times1} \\ & 1 \end{bmatrix}$$

$$\mathbb{R}^4_{(\vec{\mathbf{e}}, \underline{o})} \longleftarrow \mathbb{R}^4_{(\vec{\mathbf{e}}', \underline{o}')}$$

Quick summary / examples

- Recall: vectors
- Affine points
- Coordinate systems
- Affine transformations

- Model/camera/view
- View matrix

Positions and Displacements

Points/positions

$$\begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix}$$

Vectors/displacements

$$\begin{bmatrix} v_x \\ v_y \\ v_z \\ 0 \end{bmatrix}$$

Linear transformation on vectors

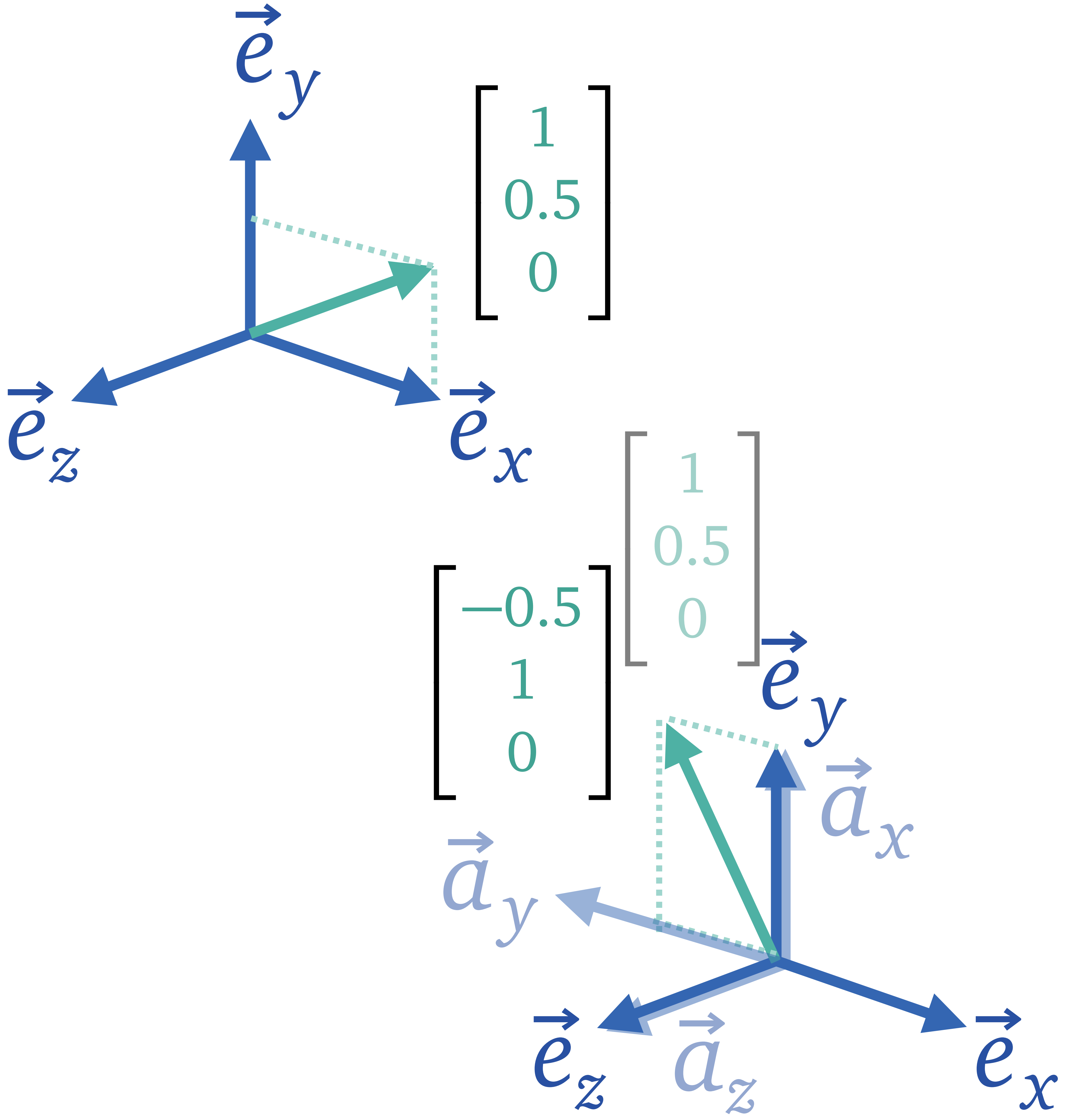
- Linear transformations are applied to vectors.
- In 3D, we don't need the 4th homogeneous coordinate.
  Just apply a 3x3 matrix to a 3D vector.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = \begin{bmatrix} a_{11}v_x + a_{12}v_y + a_{13}v_z \\ a_{21}v_x + a_{22}v_y + a_{23}v_z \\ a_{31}v_x + a_{32}v_y + a_{33}v_z \end{bmatrix}$$

Linear transformation on vectors

- Linear transformations are applied to vectors.
- In 3D, we don't need the 4th homogeneous coordinate.
  Just apply a 3x3 matrix to a 3D vector.

$$\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = \begin{bmatrix} -v_y \\ v_x \\ v_z \end{bmatrix}$$



Affine transformation on positions

- Affine transformations are applied to points.
- We need the 4th homogeneous coordinate to handle translations.

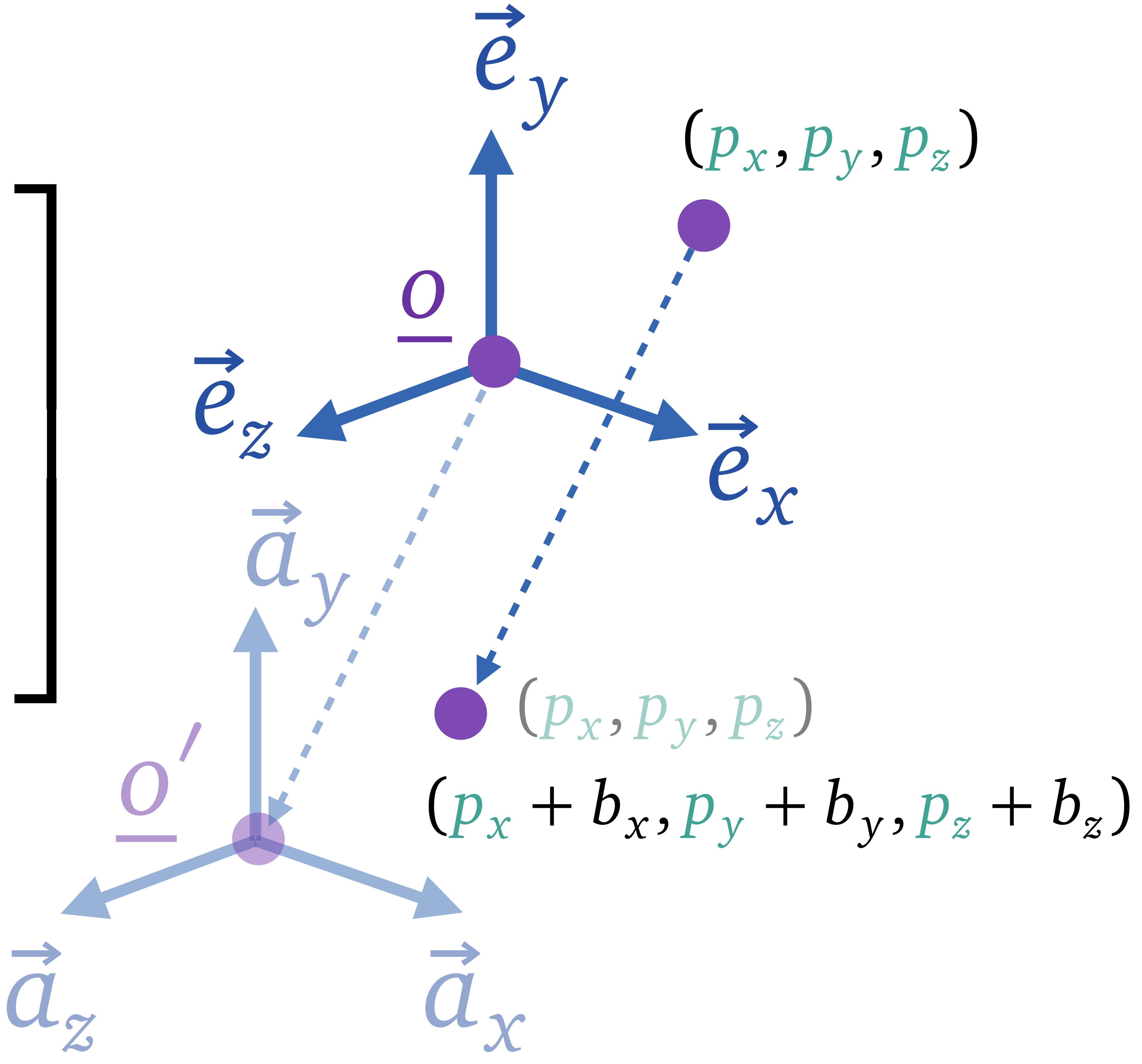
$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & b_x \\ a_{21} & a_{22} & a_{23} & b_y \\ a_{31} & a_{32} & a_{33} & b_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11}p_x + a_{12}p_y + a_{13}p_z + b_x \\ a_{21}p_x + a_{22}p_y + a_{23}p_z + b_y \\ a_{31}p_x + a_{32}p_y + a_{33}p_z + b_z \\ 1 \end{bmatrix}$$

Affine transformation on positions

- Affine transformations are applied to points.
- We need the 4th homogeneous coordinate to handle translations.

(pure translation)

$$\begin{bmatrix} 1 & & & b_x \\ & 1 & & b_y \\ & & 1 & b_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} = \begin{bmatrix} p_x + b_x \\ p_y + b_y \\ p_z + b_z \\ 1 \end{bmatrix}$$



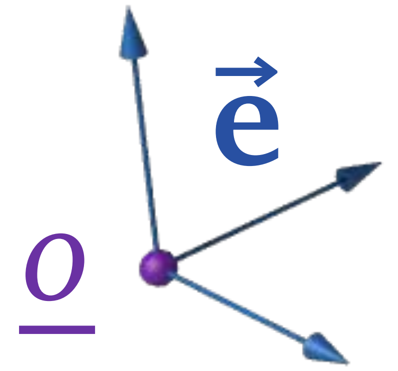
Matrix from coord systems' relation

- Exercise: Find the affine transform matrix relating the two coordinate system

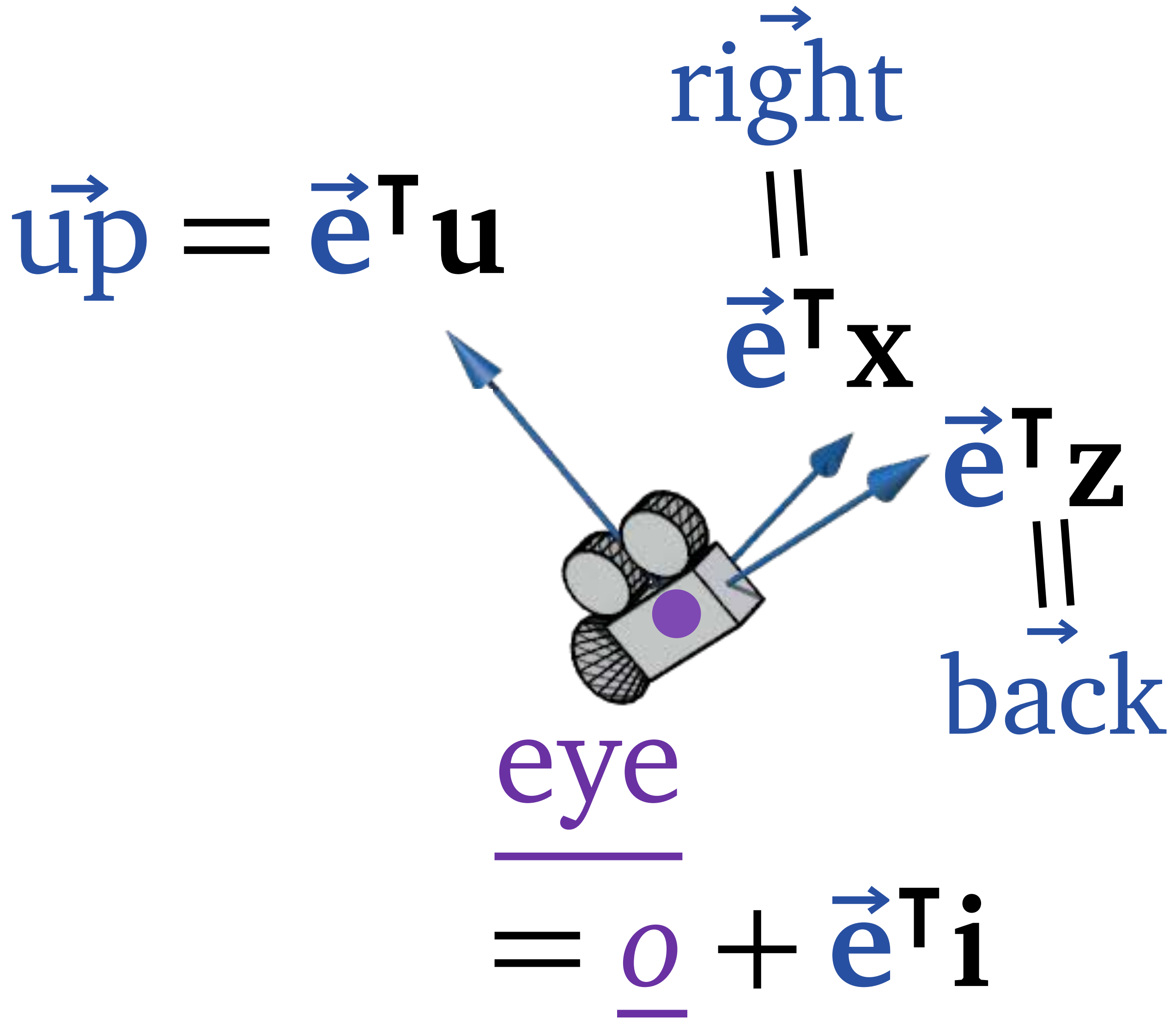
$$\begin{bmatrix}\vec{right} & \vec{up} & \vec{back} & \underline{eye}\end{bmatrix} = \begin{bmatrix}\vec{e}_1 & \vec{e}_2 & \vec{e}_3 & \underline{o}\end{bmatrix}\begin{bmatrix} & \mathbf{C} & \end{bmatrix}$$

$$\mathbb{R}^4_{\text{world}} \xleftarrow{\;\mathbf{C}\;} \mathbb{R}^4_{\text{camera}}$$

$$\mathbf{C} = \begin{bmatrix} | & | & | & | \\ \mathbf{x} & \mathbf{u} & \mathbf{z} & \mathbf{i} \\ | & | & | & | \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



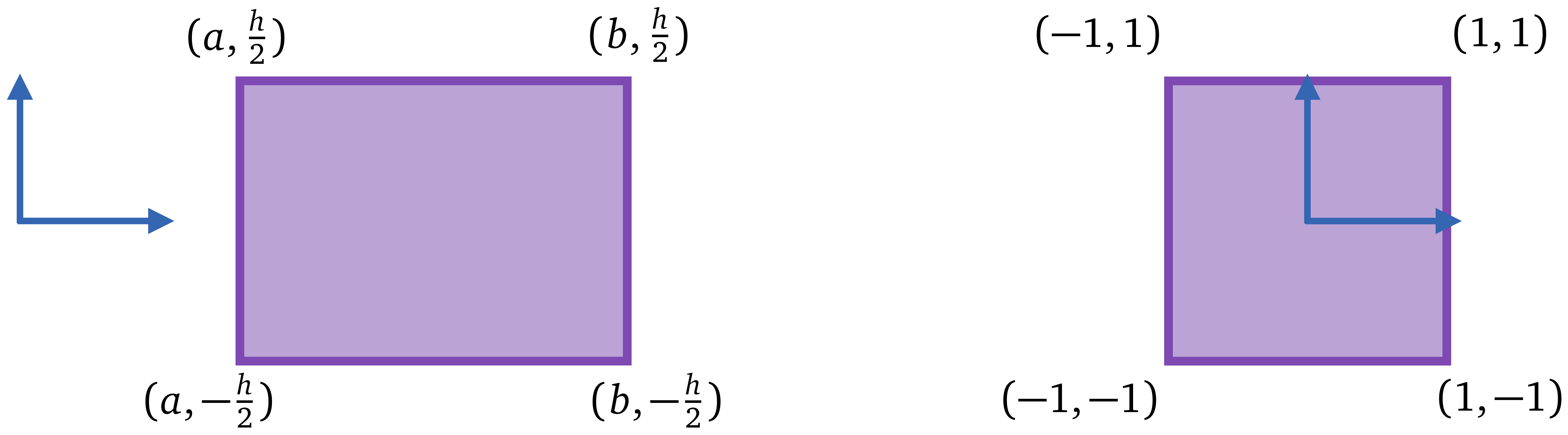
world coord system



camera coord system

Example

- Exercise: Find the transformation that transforms the following 2D box to a square centered at the origin



Example

- Exercise: Find the transformation that transforms the following 2D box to a square centered at the origin

Example

- Exercise: Find the transformation that transforms the following 2D box to a square centered at the origin
- First translate to left by $\frac{a+b}{2}$ so that the rectangle is centered about the origin
- Then scale horizontally by $\frac{2}{b-a}$ and vertically by $\frac{2}{h}$

Example

- First translate to left by $\frac{a+b}{2}$ so that the rectangle is centered about the origin

$$\mathbf{T} = \begin{bmatrix} 1 & & -\frac{a+b}{2} \\ & 1 & 0 \\ & & 1 \end{bmatrix}$$

- Then scale horizontally by $\frac{2}{b-a}$ and vertically by $\frac{2}{h}$

$$\mathbf{S} = \begin{bmatrix} \frac{2}{b-a} & & 0 \\ & \frac{2}{h} & 0 \\ & & 1 \end{bmatrix}$$

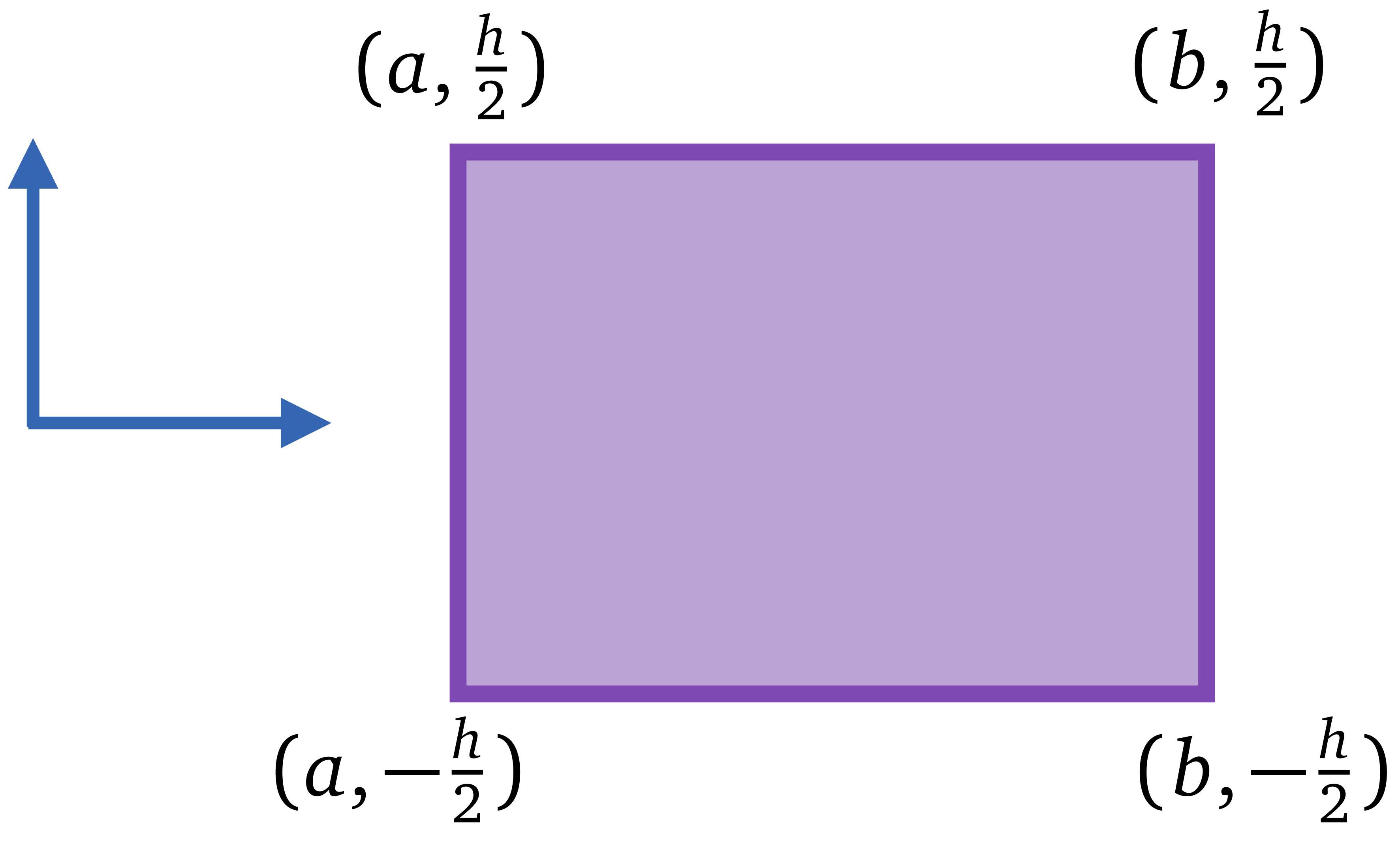
Example

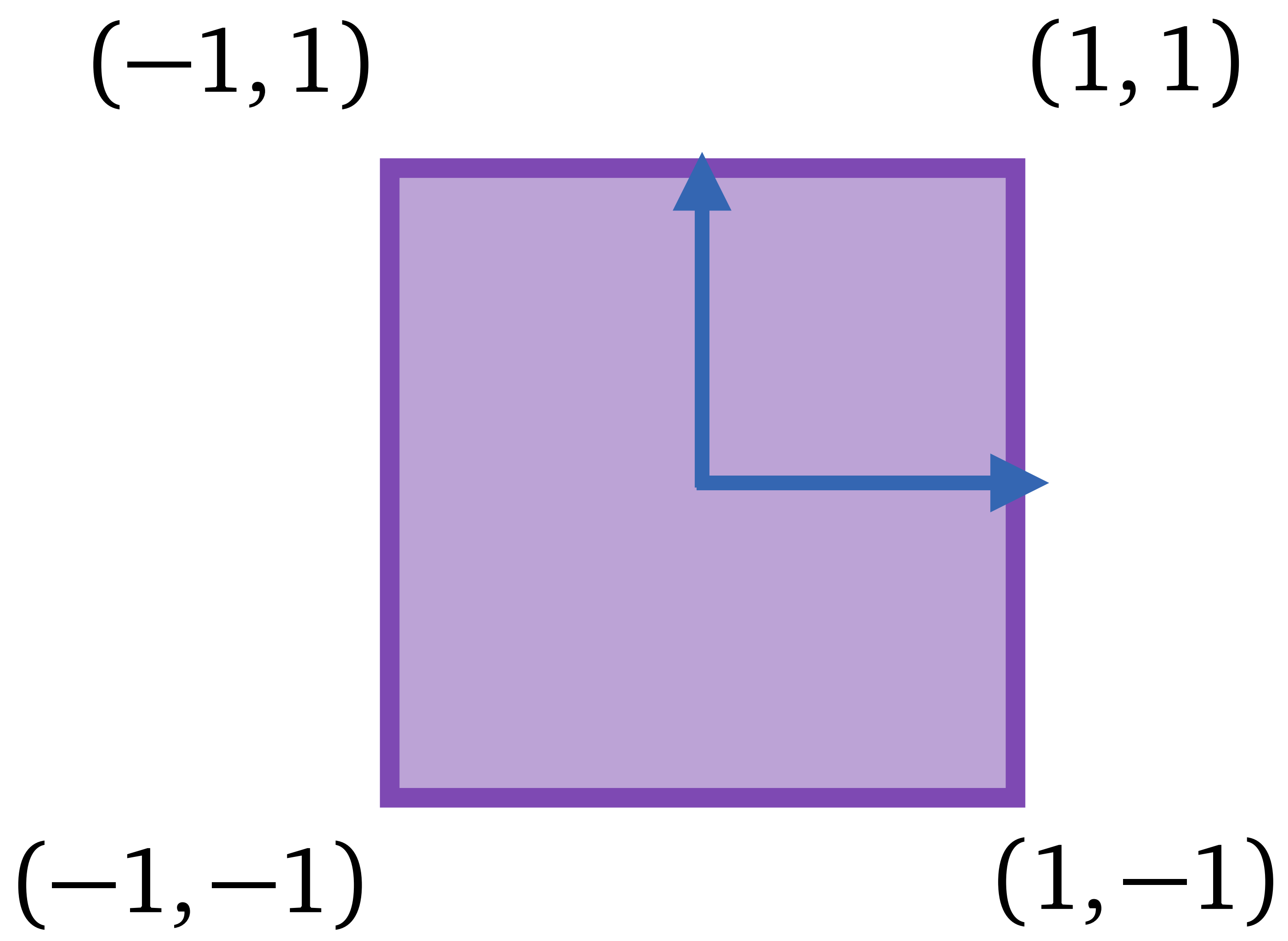
$$\mathbf{T} = \begin{bmatrix} 1 & & -\frac{a+b}{2} \\ & 1 & 0 \\ & & 1 \end{bmatrix} \qquad \mathbf{S} = \begin{bmatrix} \frac{2}{b-a} & & 0 \\ & \frac{2}{h} & 0 \\ & & 1 \end{bmatrix}$$

$$\mathbf{ST} = \begin{bmatrix} \frac{2}{b-a} & & 0 \\ & \frac{2}{h} & 0 \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & -\frac{a+b}{2} \\ & 1 & 0 \\ & & 1 \end{bmatrix} = \begin{bmatrix} \frac{2}{b-a} & & -\frac{a+b}{b-a} \\ & \frac{2}{h} & 0 \\ & & 1 \end{bmatrix}$$

Example

$$\mathbf{ST} = \begin{bmatrix} \frac{2}{b-a} & & 0 \\ & \frac{2}{h} & 0 \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & -\frac{a+b}{2} \\ & 1 & 0 \\ & & 1 \end{bmatrix} = \begin{bmatrix} \frac{2}{b-a} & & -\frac{a+b}{b-a} \\ & \frac{2}{h} & 0 \\ & & 1 \end{bmatrix}$$





Transformations in Graphics

- Recall: vectors
- Affine points
- Coordinate systems
- Affine transformations
- Model/camera/view
- View matrix

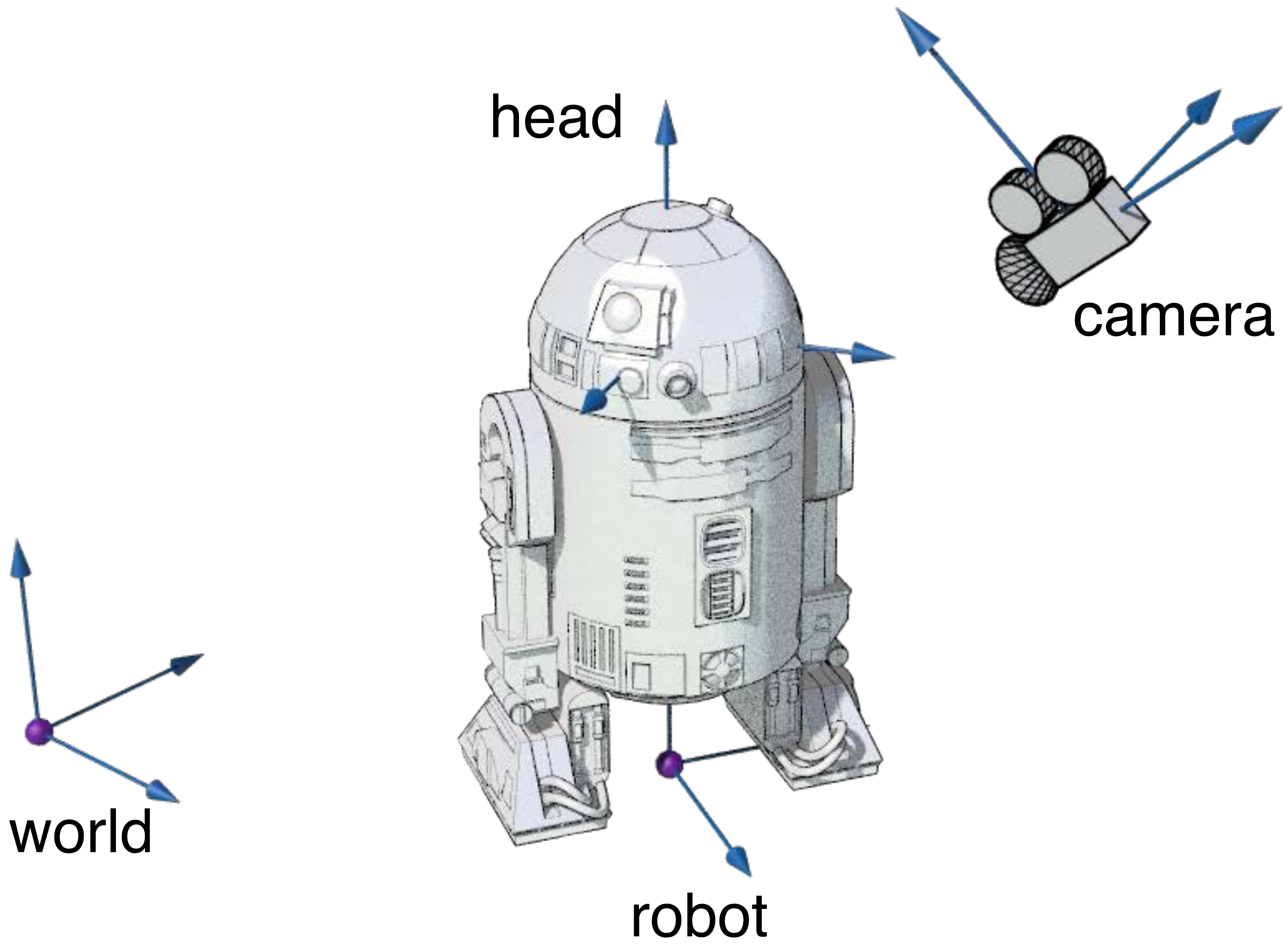
In practice

- Positions are stored as 4D array of numbers with the last entry = 1.
- Each of the 4D array of numbers correspond to a geometric position under a frame.
- Between two different frames, we record a 4-by-4 matrix that takes the form

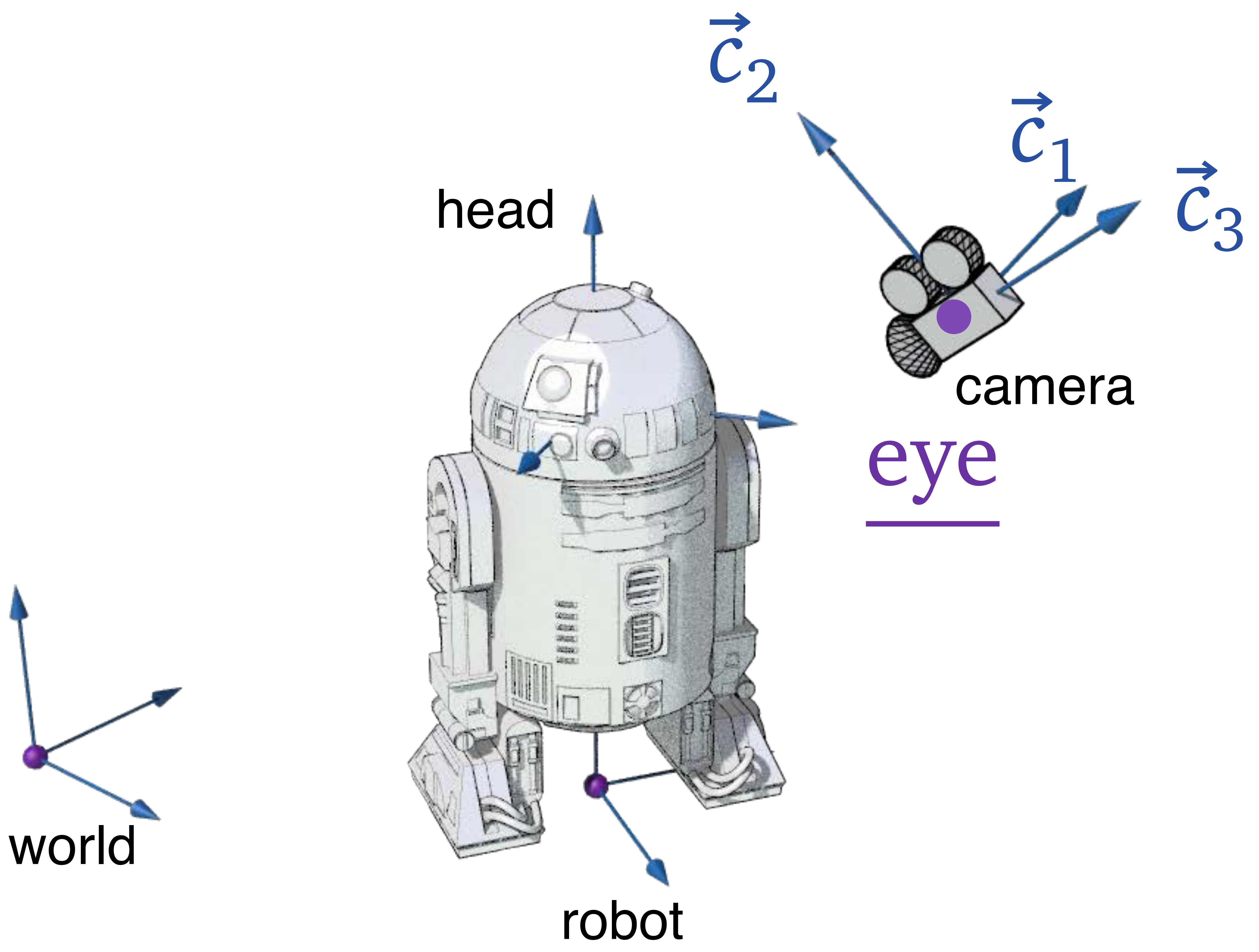
$$\mathbf{M} = \begin{bmatrix} \mathbf{A}_{3\times 3} & \mathbf{b}_{3\times 1} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}$$

this is the “ratio” between two coordinate systems.

A typical scene



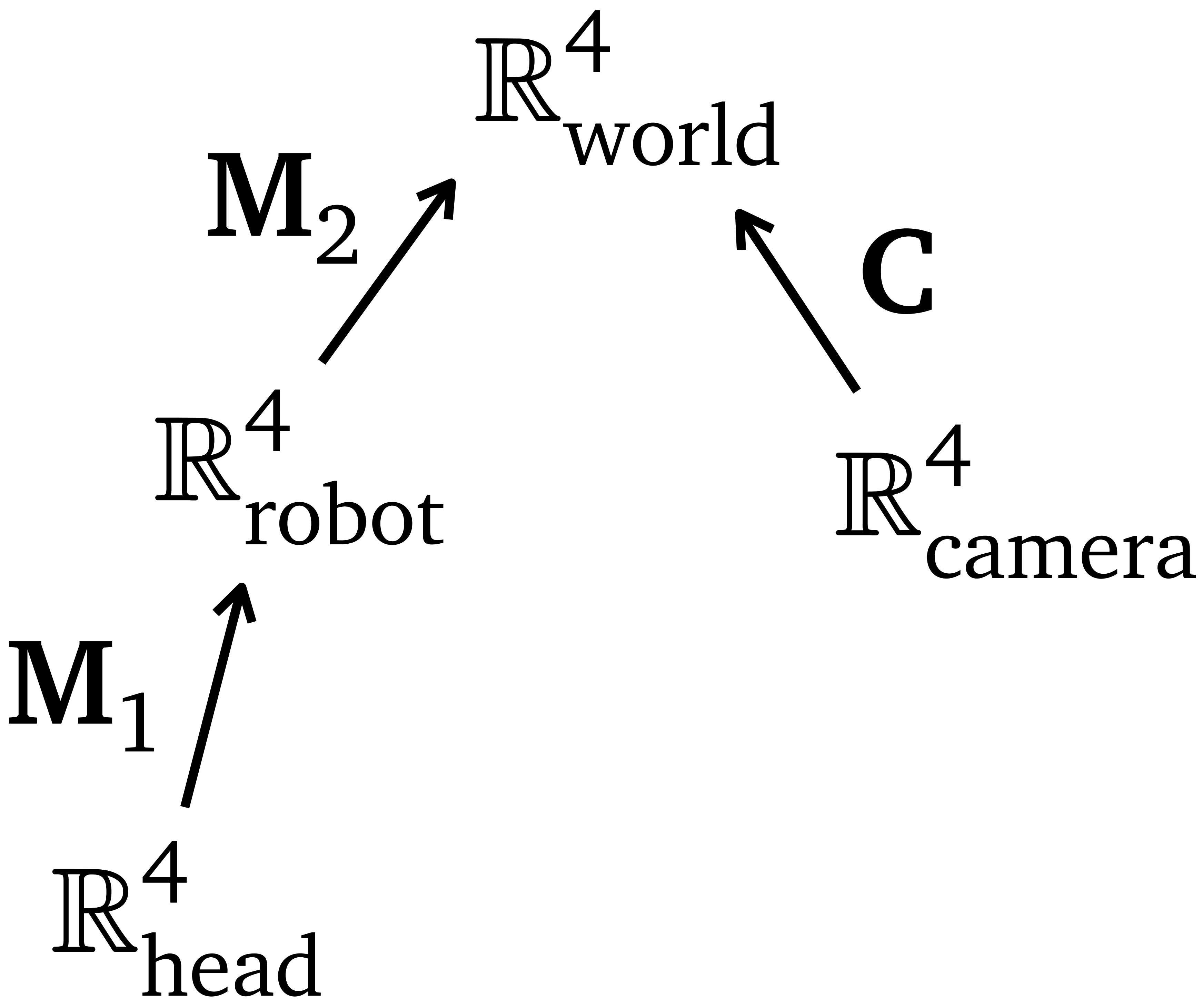
Camera's coordinate system

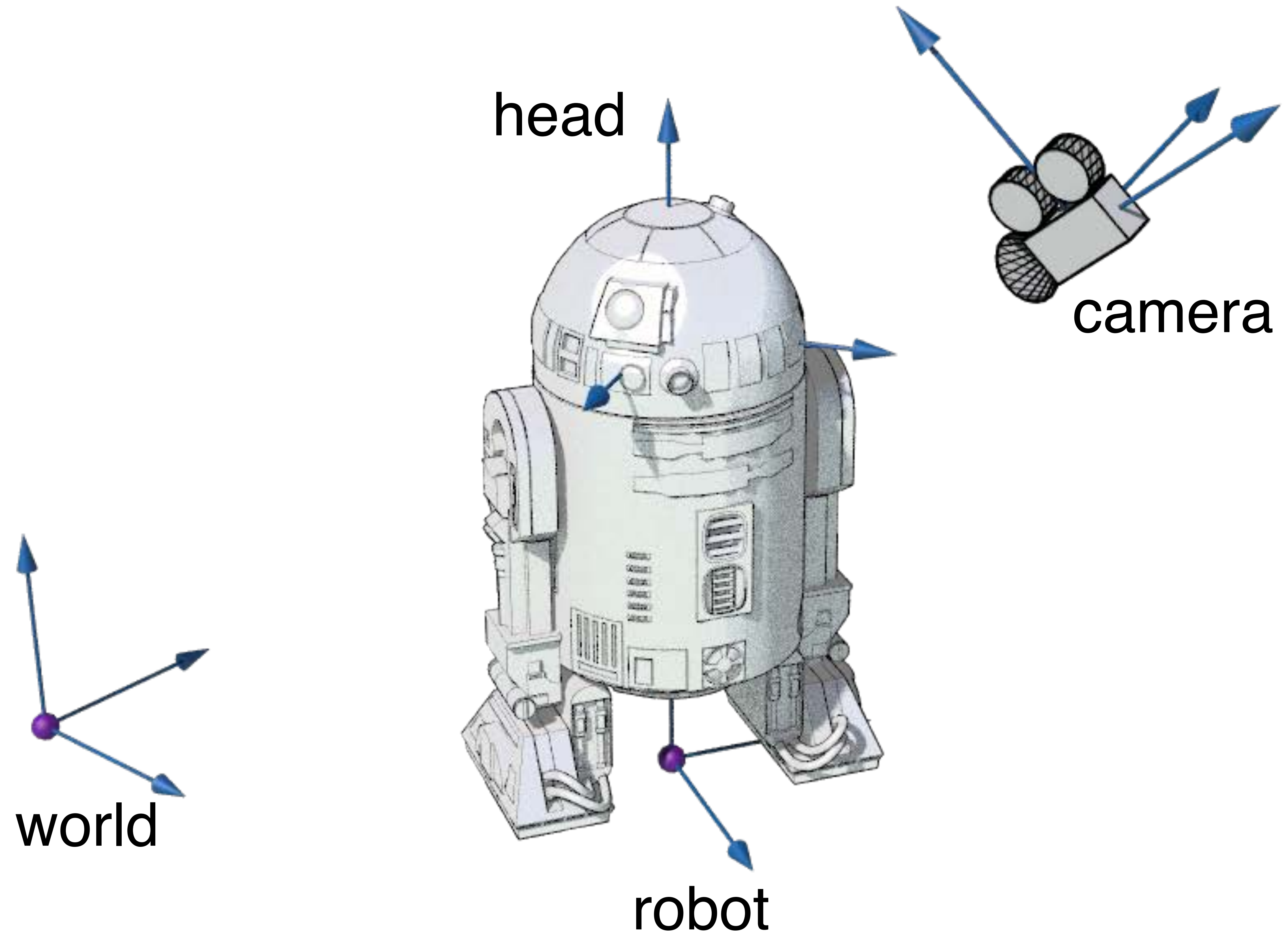


$[\vec{c}_1 \quad \vec{c}_2 \quad \vec{c}_3 \quad \underline{\text{eye}}]$

- $\vec{c}_1$, camera's x vector, points to the right of the camera.
- $\vec{c}_2$, camera's up vector, points to the top of the camera.
- $\vec{c}_3$, camera's z vector, points to the back of the camera.
- $\underline{\text{eye}}$ is camera's position.

A typical scene





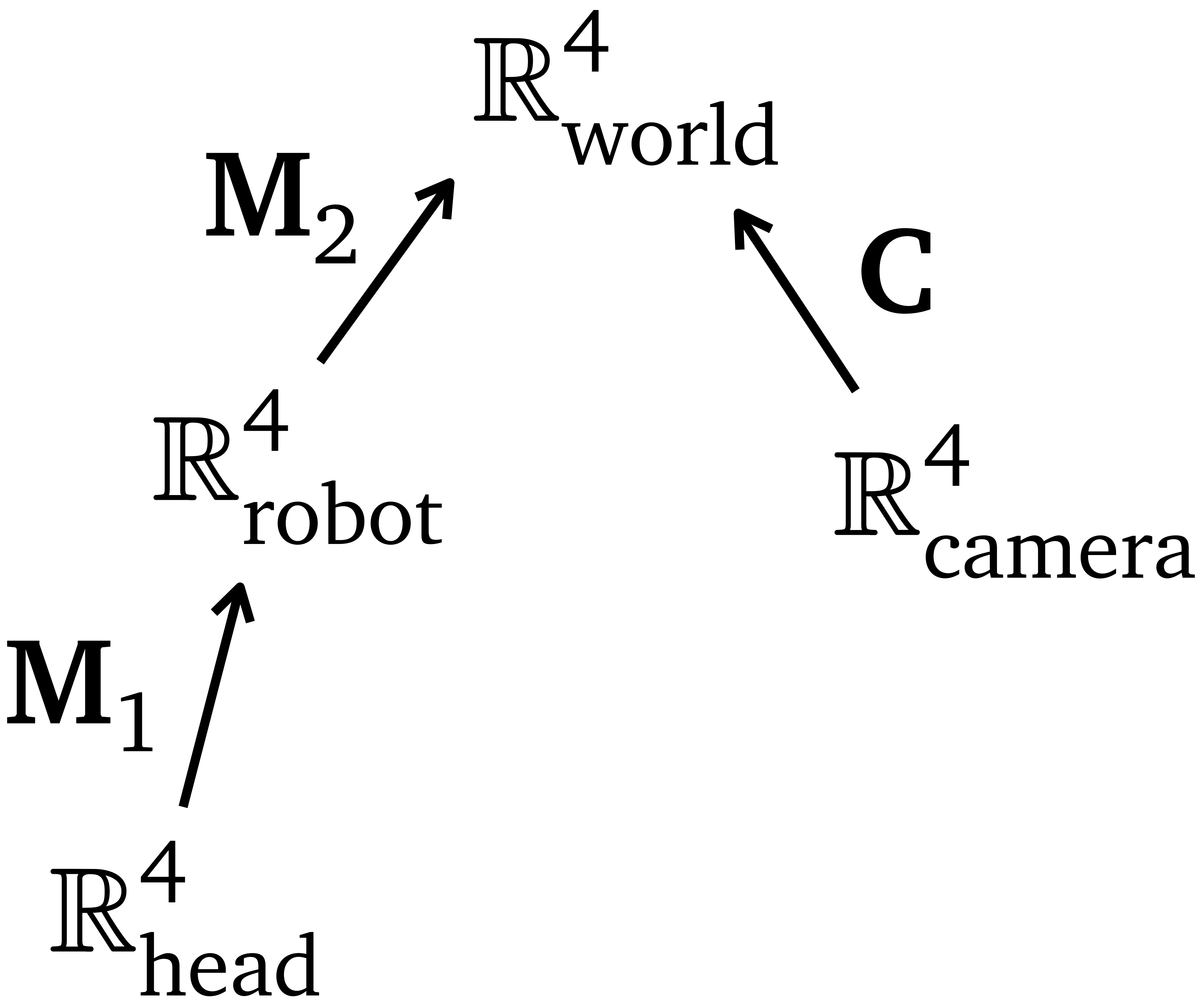
Terminologies

- The 4x4 matrices $\mathbf{M}_1, \mathbf{M}_2$ (from model towards the world) are called **model matrices**.

- The model matrix $\mathbf{C}$ of the camera is called the **camera matrix**.

- The inverse of the camera matrix $\mathbf{V} = \mathbf{C}^{-1}$ is called the **view matrix**.

- The matrix we apply to the 4D array of number in the "head" coordinate is $\mathbf{V}\mathbf{M}_2\mathbf{M}_1$ called the **model-view matrix**.



Matrices are multiplied in the v-shader

vertex shader

```
# version 330 core

layout (location = 0) in vec3 vertex_position;
layout (location = 1) in vec3 vertex_normal;

uniform mat4 modelview;
uniform mat4 projection;

out vec4 position;
out vec3 normal;

void main(){
    gl_Position = projection * modelview * vec4( vertex_position, 1.0f );

    // forward the raw position and normal to frag shader
    position = vec4(vertex_position, 1.0f );
    normal = vertex_normal;
}
```

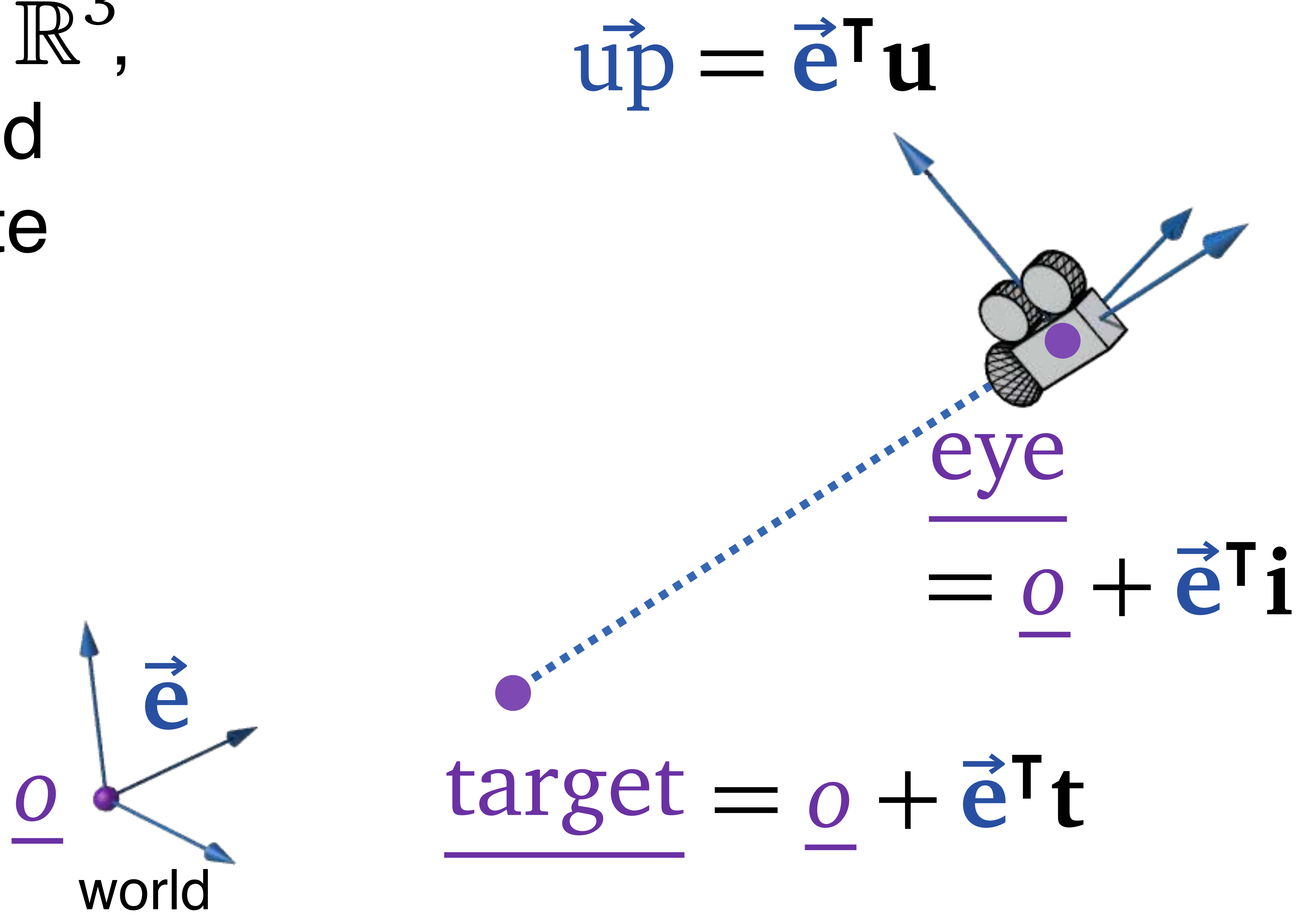
Compute the View Matrix

- Recall: vectors
- Affine points
- Coordinate systems
- Affine transformations
- Model/camera/view
- View matrix

Compute the view matrix

Task

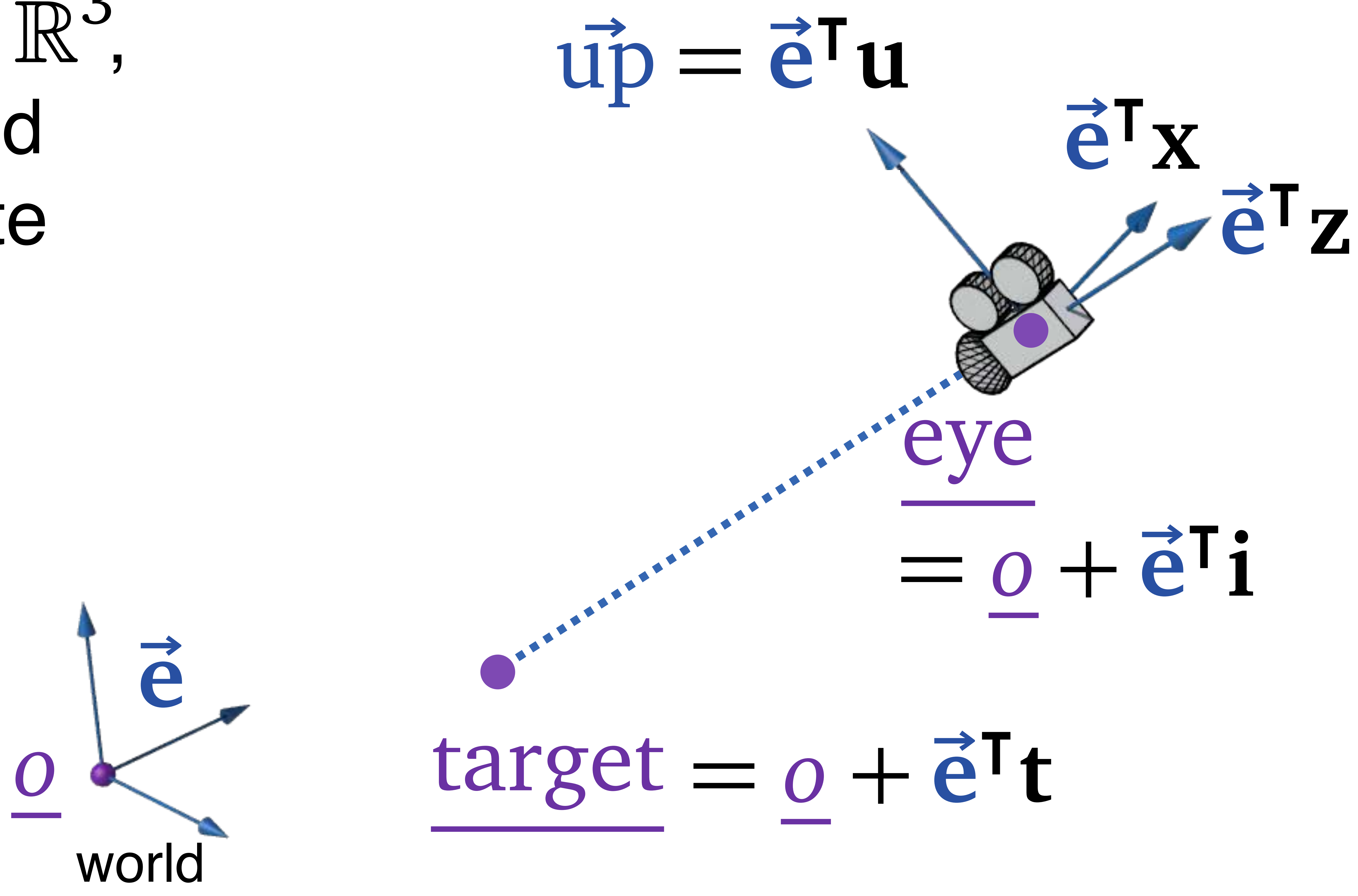
Given the eye position $\mathbf{i} \in \mathbb{R}^3$, target position $\mathbf{t} \in \mathbb{R}^3$, and up-vector $\mathbf{u} \in \mathbb{R}^3$, compute the 4x4 view matrix $\mathbf{V}$.



Compute the view matrix

Task

Given the eye position $\mathbf{i} \in \mathbb{R}^3$, target position $\mathbf{t} \in \mathbb{R}^3$, and up-vector $\mathbf{u} \in \mathbb{R}^3$, compute the 4x4 view matrix $\mathbf{V}$.

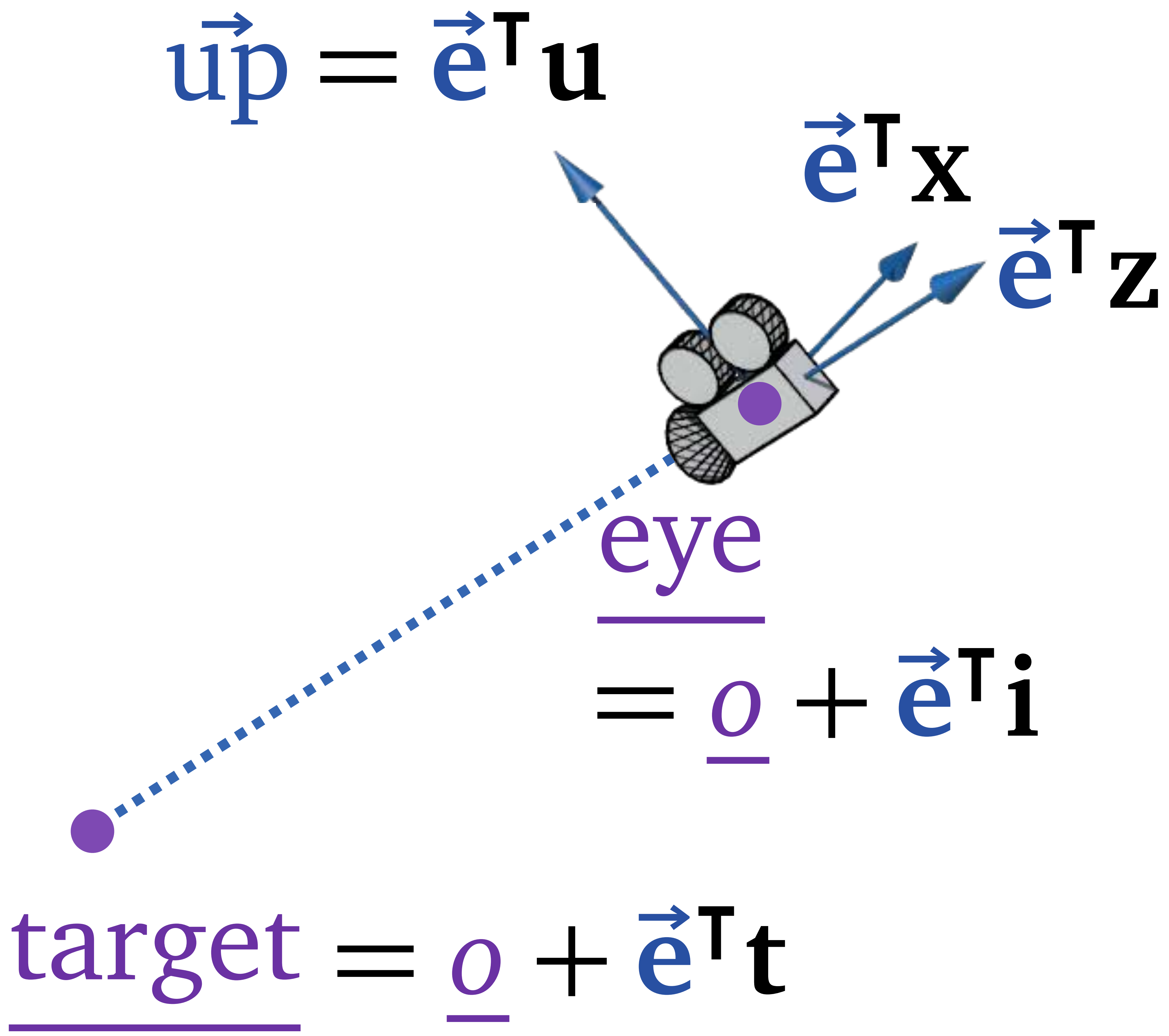


Compute the view matrix

Step 1

Compute $\mathbf{z}$:

$$\mathbf{z} = \frac{\mathbf{i} - \mathbf{t}}{|\mathbf{i} - \mathbf{t}|}$$



Compute the view matrix

Step 1

Compute $\mathbf{z}$:

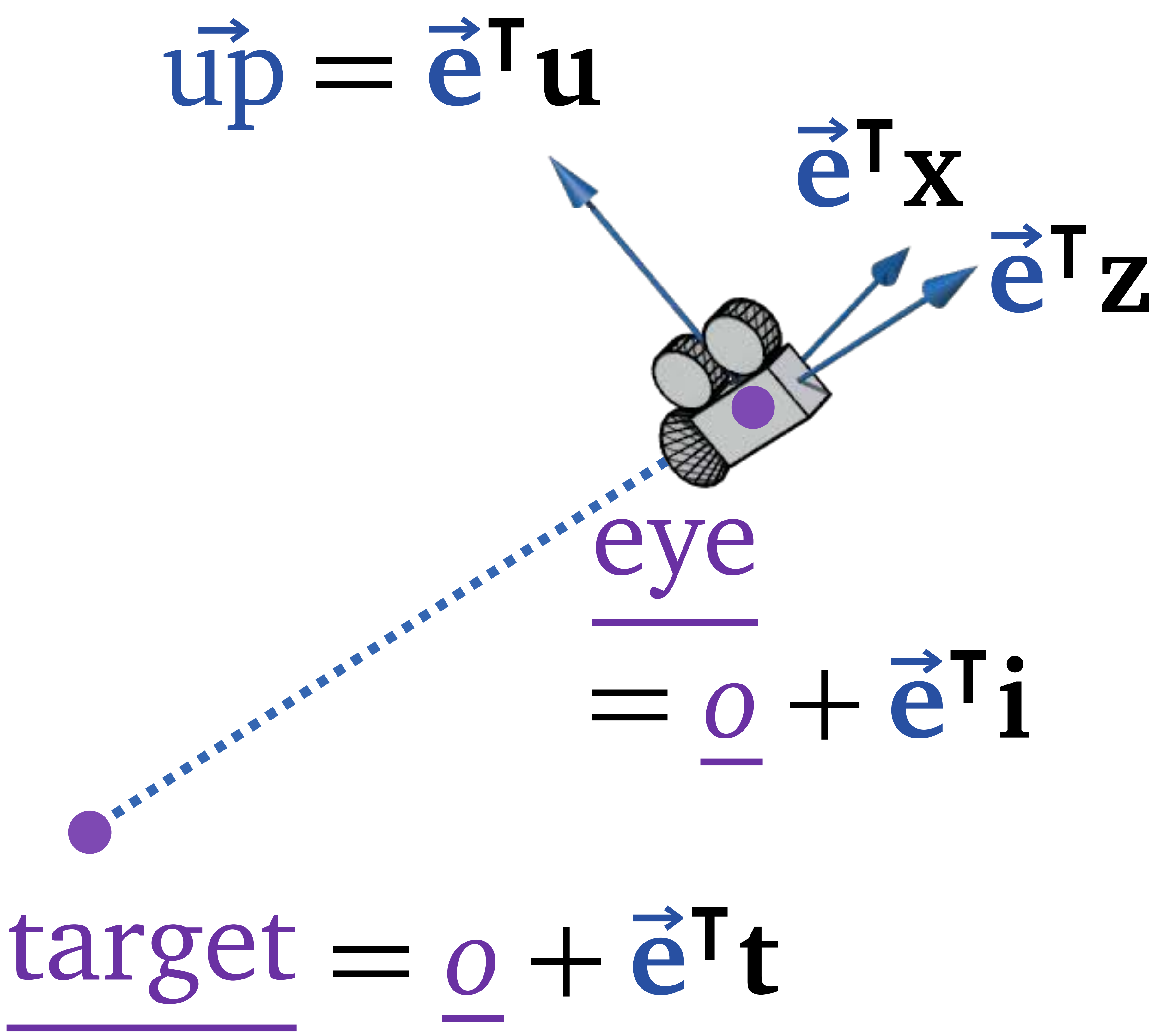
$$\mathbf{z} = \frac{\mathbf{i} - \mathbf{t}}{|\mathbf{i} - \mathbf{t}|}$$

Step 2

Ensure that $\mathbf{u}$ is orthogonal to $\mathbf{z}$

$$\mathbf{u} \leftarrow \mathbf{u} - (\mathbf{z} \cdot \mathbf{u})\mathbf{z}$$

$$\mathbf{u} \leftarrow \frac{\mathbf{u}}{|\mathbf{u}|}$$



Compute the view matrix

Step 2

Ensure that $\mathbf{u}$ is orthogonal to $\mathbf{z}$

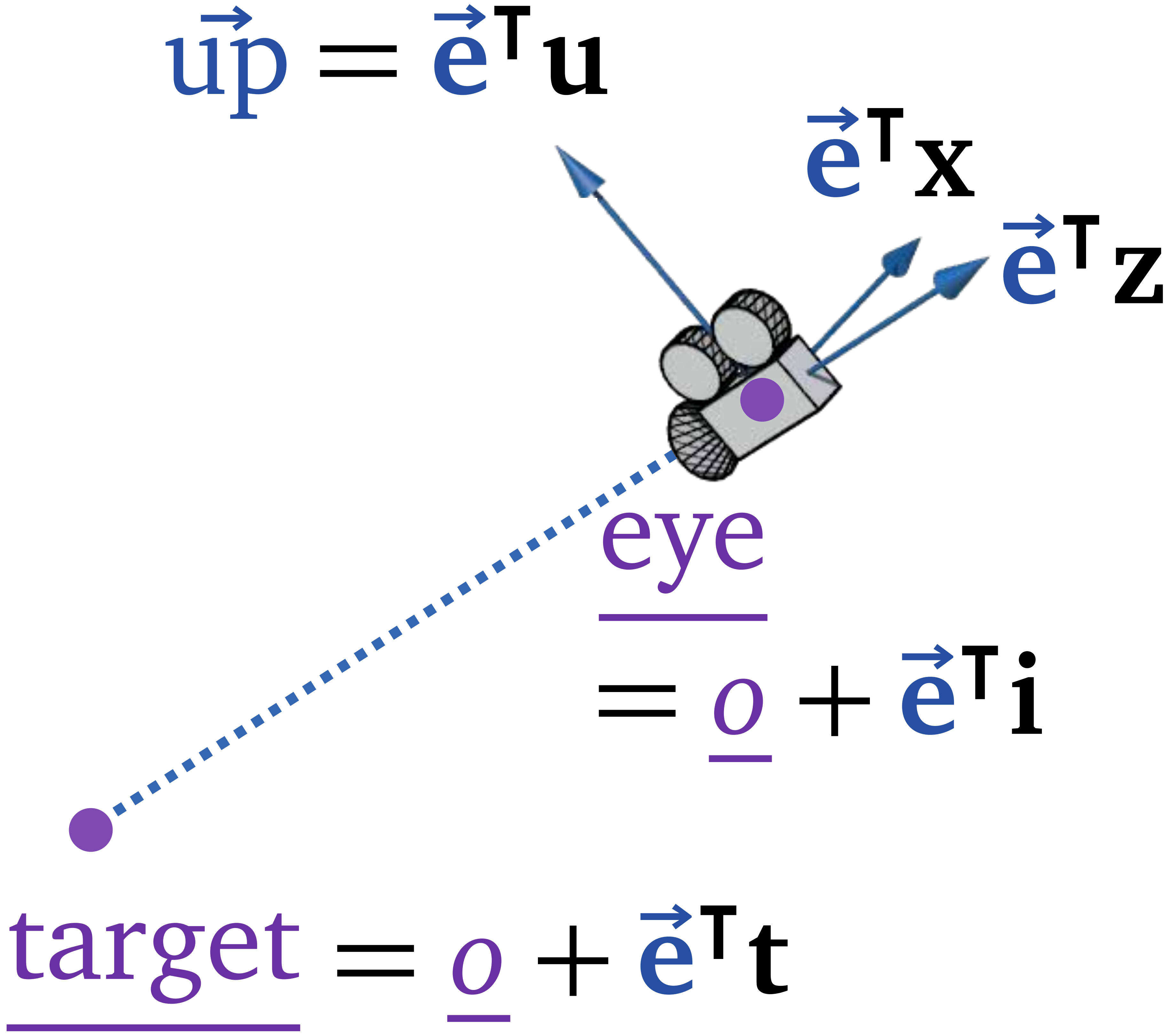
$$\mathbf{u} \leftarrow \mathbf{u} - (\mathbf{z} \cdot \mathbf{u})\mathbf{z}$$

$$\mathbf{u} \leftarrow \frac{\mathbf{u}}{|\mathbf{u}|}$$

Step 3

Compute $\mathbf{x}$:

$$\mathbf{x} = \mathbf{u} \times \mathbf{z}$$



Compute the view matrix

Step 3

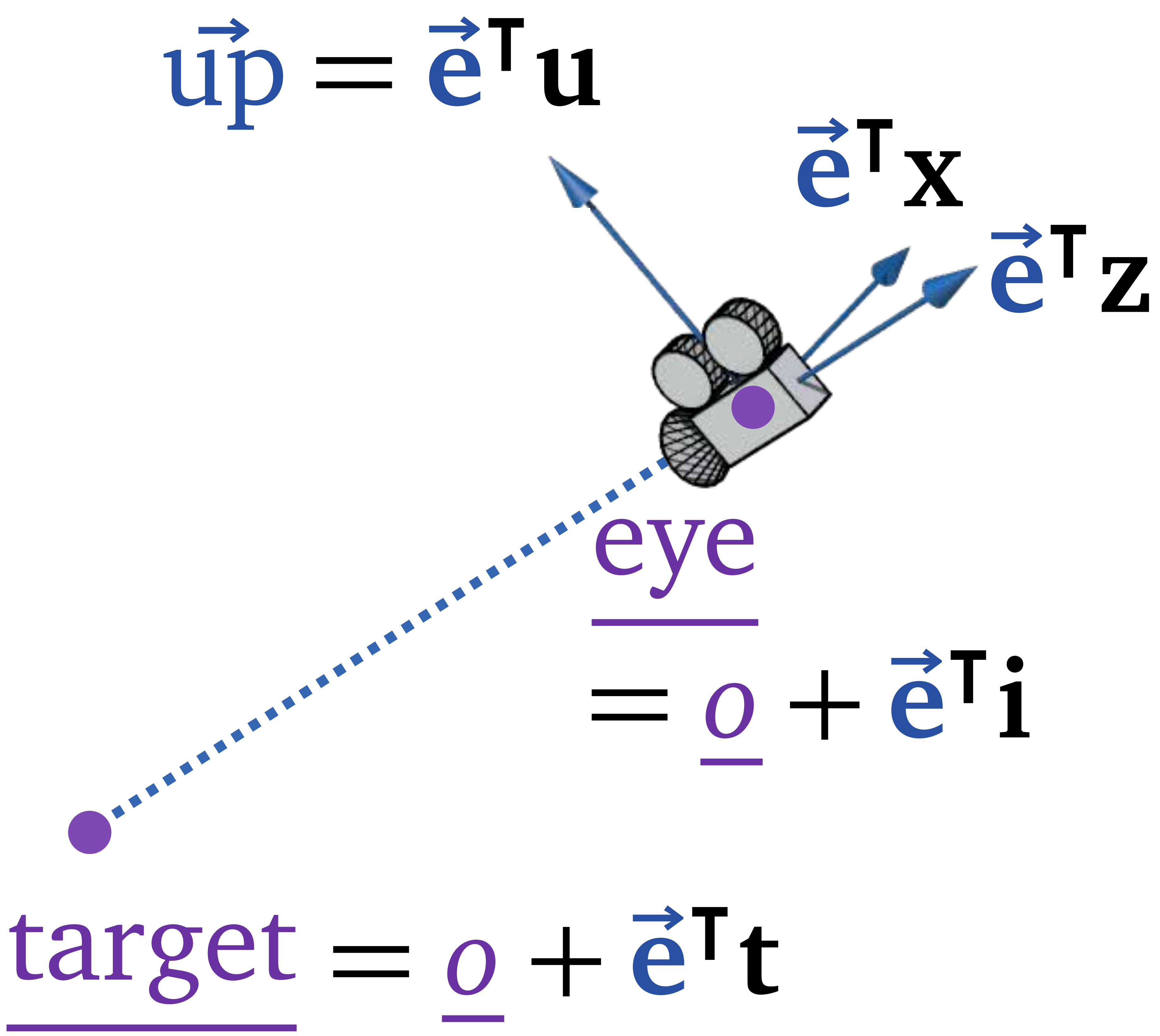
Compute $\mathbf{x}$:

$$\mathbf{x} = \mathbf{u} \times \mathbf{z}$$

Step 4

Camera matrix (model matrix for the camera)

$$\mathbf{C} = \begin{bmatrix} | & | & | & | \\ \mathbf{x} & \mathbf{u} & \mathbf{z} & \mathbf{i} \\ | & | & | & | \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Compute the view matrix

Step 4

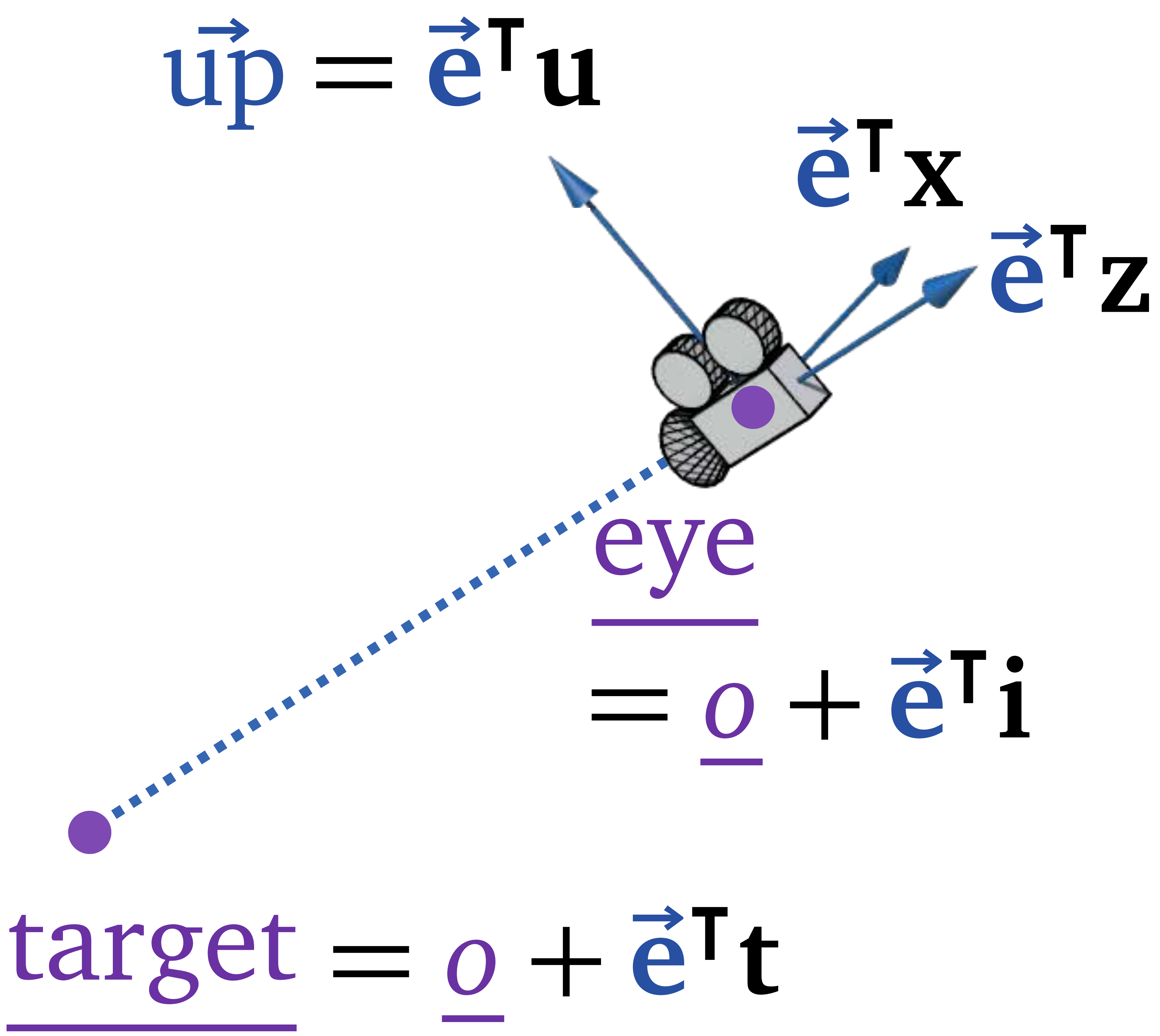
Camera matrix (model matrix for the camera)

$$\mathbf{C} = \begin{bmatrix} | & | & | & | \\ \mathbf{x} & \mathbf{u} & \mathbf{z} & \mathbf{i} \\ | & | & | & | \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Final Step

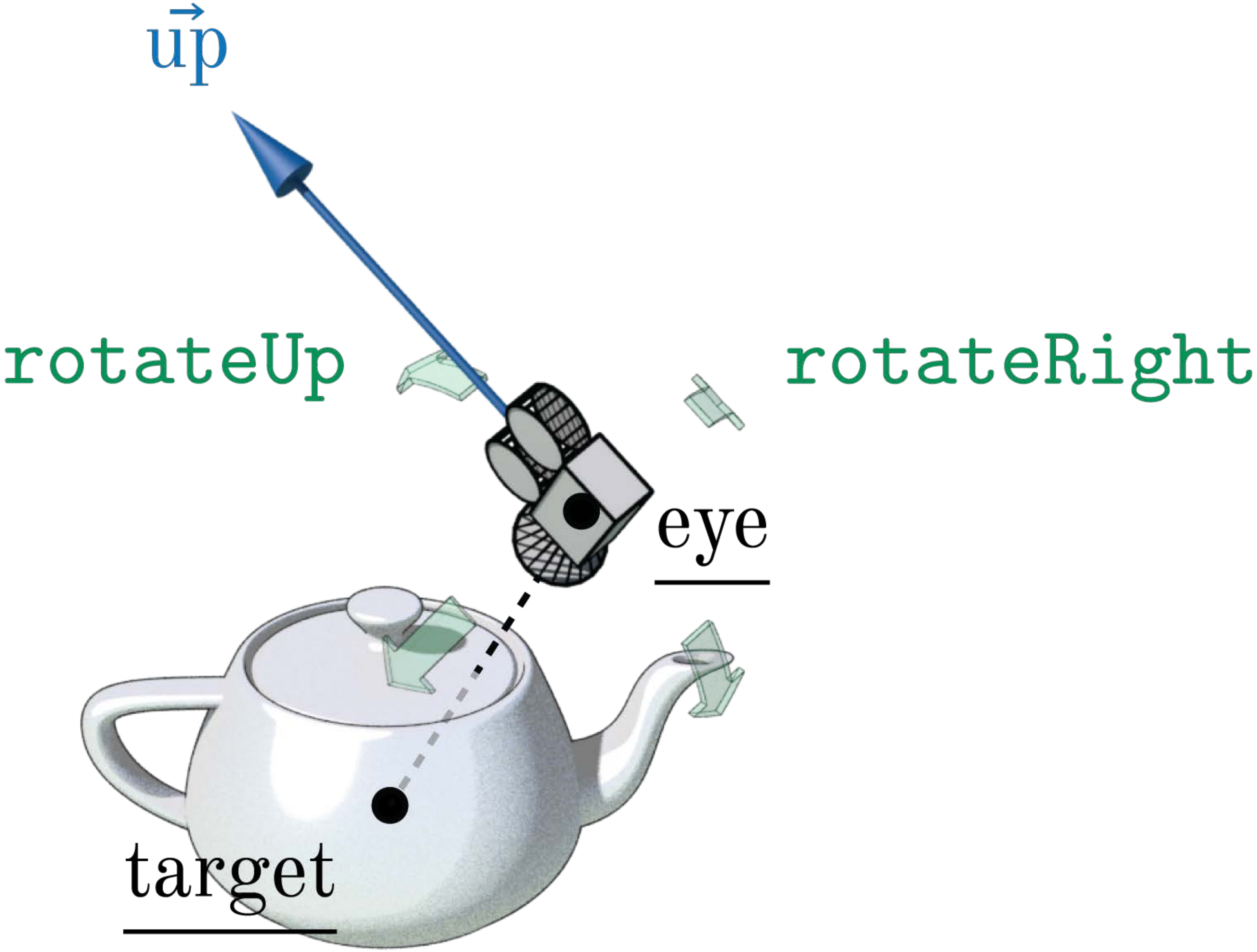
$$\mathbf{V} = \mathbf{C}^{-1}$$

(you can use `glm::inverse`)



Miscellaneous

Camera::rotateRight, rotateUp



Induced transformation

Induced transformation

Induced transformation

- If all positions of a geometric object is transformed by an affine transformation

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & b_x \\ a_{21} & a_{22} & a_{23} & b_y \\ a_{31} & a_{32} & a_{33} & b_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11}p_x + a_{12}p_y + a_{13}p_z + b_x \\ a_{21}p_x + a_{22}p_y + a_{23}p_z + b_y \\ a_{31}p_x + a_{32}p_y + a_{33}p_z + b_z \\ 1 \end{bmatrix}$$

- Then all displacement vectors are transformed by the upper-left 3x3 block matrix

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = \begin{bmatrix} a_{11}v_x + a_{12}v_y + a_{13}v_z \\ a_{21}v_x + a_{22}v_y + a_{23}v_z \\ a_{31}v_x + a_{32}v_y + a_{33}v_z \end{bmatrix}$$

Induced transformation

- If all positions of a geometric object is transformed by an affine transformation

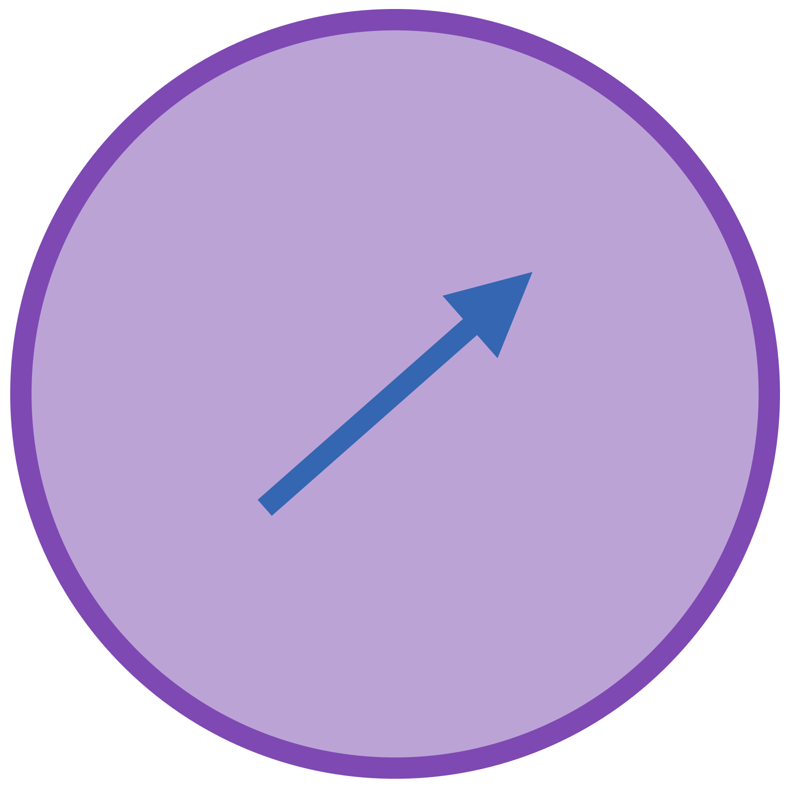
$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & b_x \\ a_{21} & a_{22} & a_{23} & b_y \\ a_{31} & a_{32} & a_{33} & b_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11}p_x + a_{12}p_y + a_{13}p_z + b_x \\ a_{21}p_x + a_{22}p_y + a_{23}p_z + b_y \\ a_{31}p_x + a_{32}p_y + a_{33}p_z + b_z \\ 1 \end{bmatrix}$$

- How about normal vectors?

Induced Transformation on Vectors

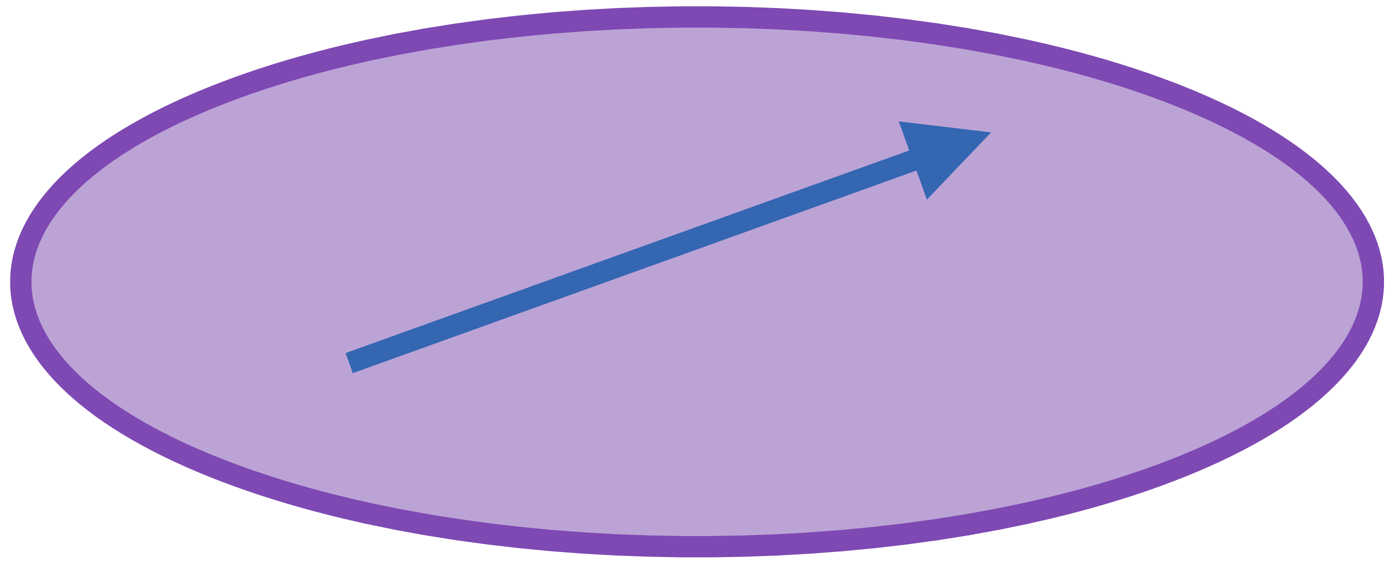
- Suppose we have an affine transformation on positions

$$\begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix} \mapsto \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix}$$



- Then displacement vectors will transform by

$$\begin{bmatrix} \mathbf{u} \end{bmatrix} \mapsto \begin{bmatrix} \mathbf{A} \end{bmatrix} \begin{bmatrix} \mathbf{u} \end{bmatrix}$$



Induced Transformation on Normals

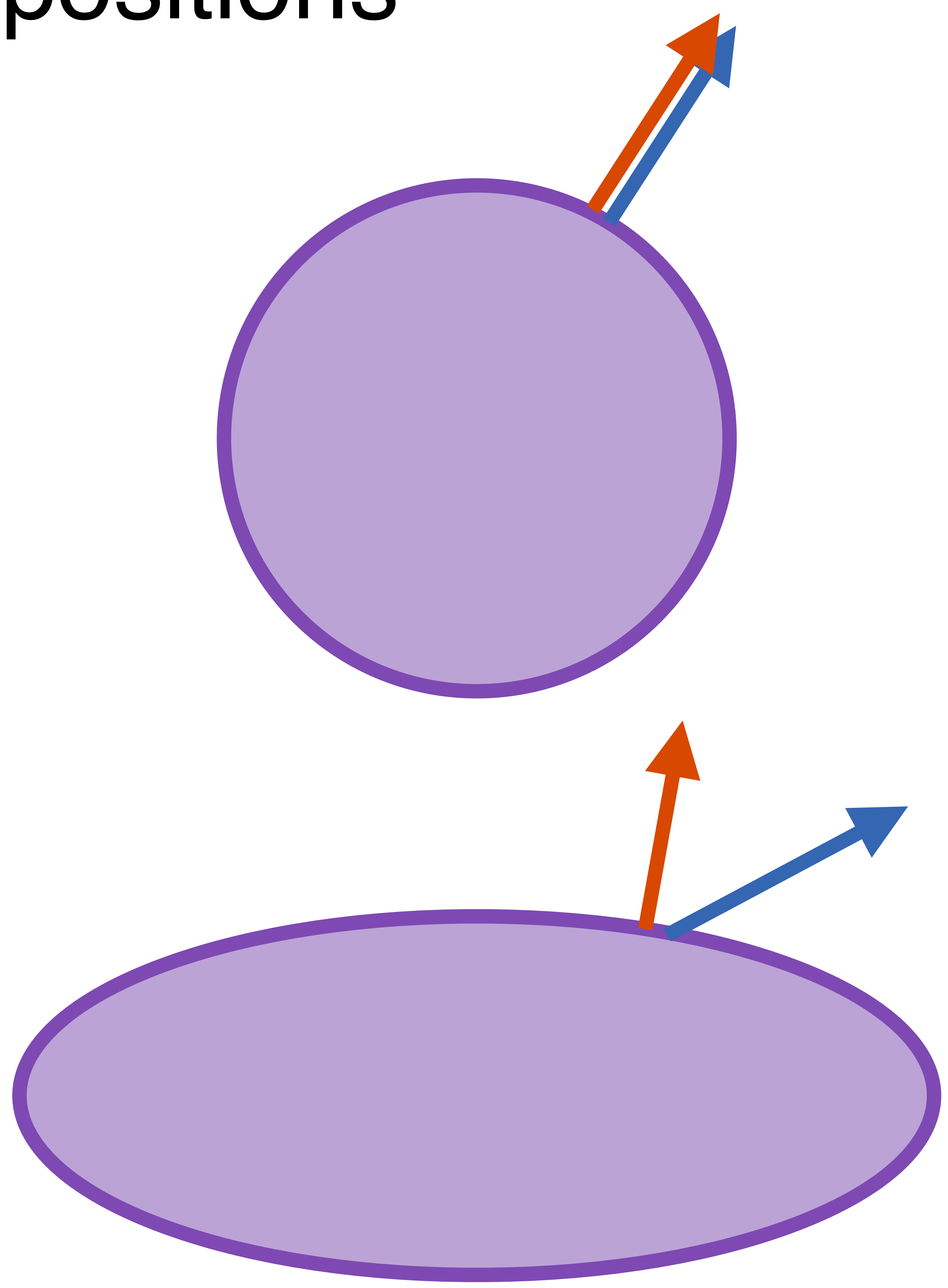
- Suppose we have an affine transformation on positions

$$\begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix} \mapsto \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix}$$

- and normal vectors transform according to

$$\begin{bmatrix} \mathbf{n} \end{bmatrix} \mapsto \begin{bmatrix} \text{What is this 3x3 matrix?} \end{bmatrix} \begin{bmatrix} \mathbf{n} \end{bmatrix}$$

(followed by a normalization)



Induced Transformation on Normals

- Suppose we have an affine transformation on positions

$$\begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix} \mapsto \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix}$$

- and normal vectors transform according to

$$\begin{bmatrix} \mathbf{n} \end{bmatrix} \mapsto \begin{bmatrix} \mathbf{A}^{-\mathsf{T}} \end{bmatrix} \begin{bmatrix} \mathbf{n} \end{bmatrix}$$

(followed by a normalization)