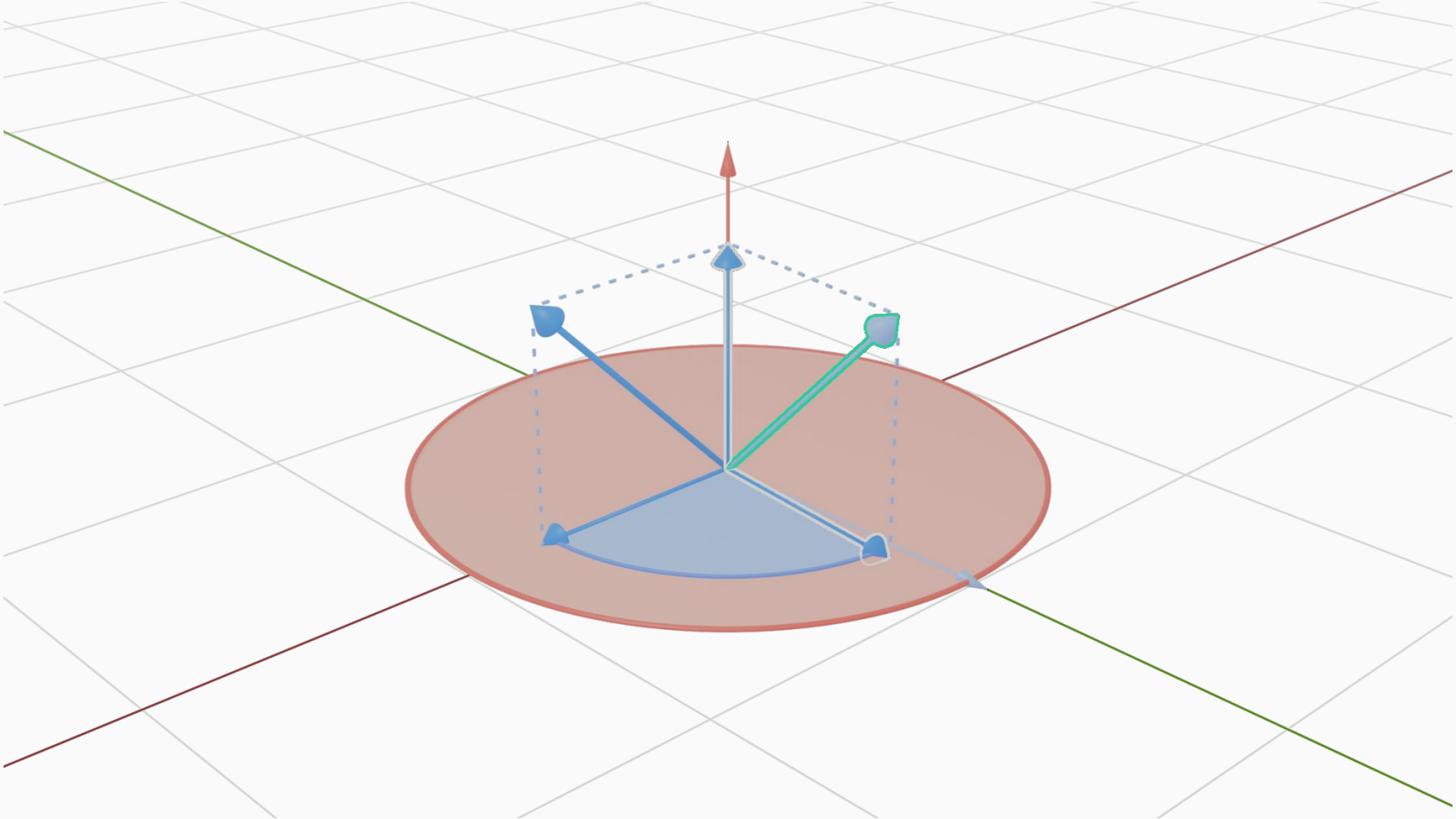
CSE 167: Computer Graphics

Discussion - Week 3

Upcoming Deadlines

Week	10/17: Hierarchical modeling	10/19: Hierarchical modeling	10/21: Lighting
4		• Exercise 4 • **Programming Assignment 2**	



Recap: Rodrigues Formula

In `Camera.cpp::rotation()`:

$$\mathbf{R}^{\mathbf{a},\theta} = \cos\theta\ \mathbf{I} + (1 - \cos\theta)\mathbf{a}\mathbf{a}^\intercal + \sin\theta\ [\mathbf{a}\times]$$

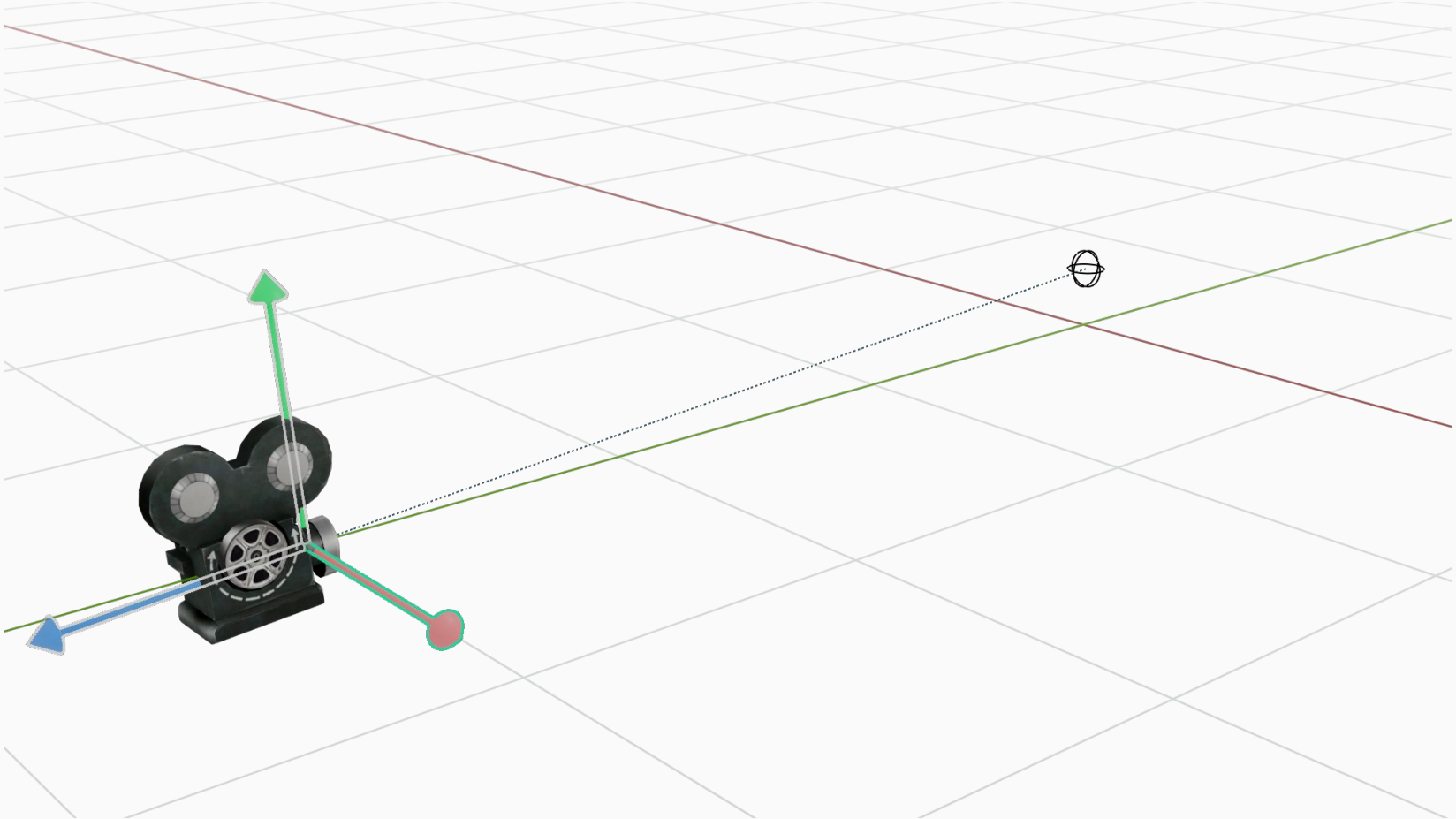
$\mathbf{a}\mathbf{a}^\intercal$ = `glm::outerProduct(a, a)`

$[\mathbf{a}\times]$ = `glm::`**`transpose`**`(`

$$\begin{bmatrix} 0 & -\mathbf{a}_z & \mathbf{a}_y \\ \mathbf{a}_z & 0 & -\mathbf{a}_x \\ -\mathbf{a}_y & \mathbf{a}_x & 0 \end{bmatrix}$$

`)`

Why?
Because `glm` is **column-major**!

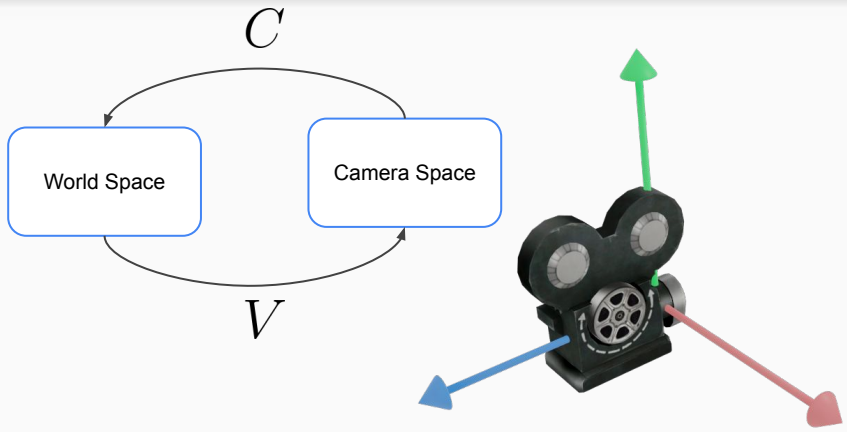


How to build the view matrix

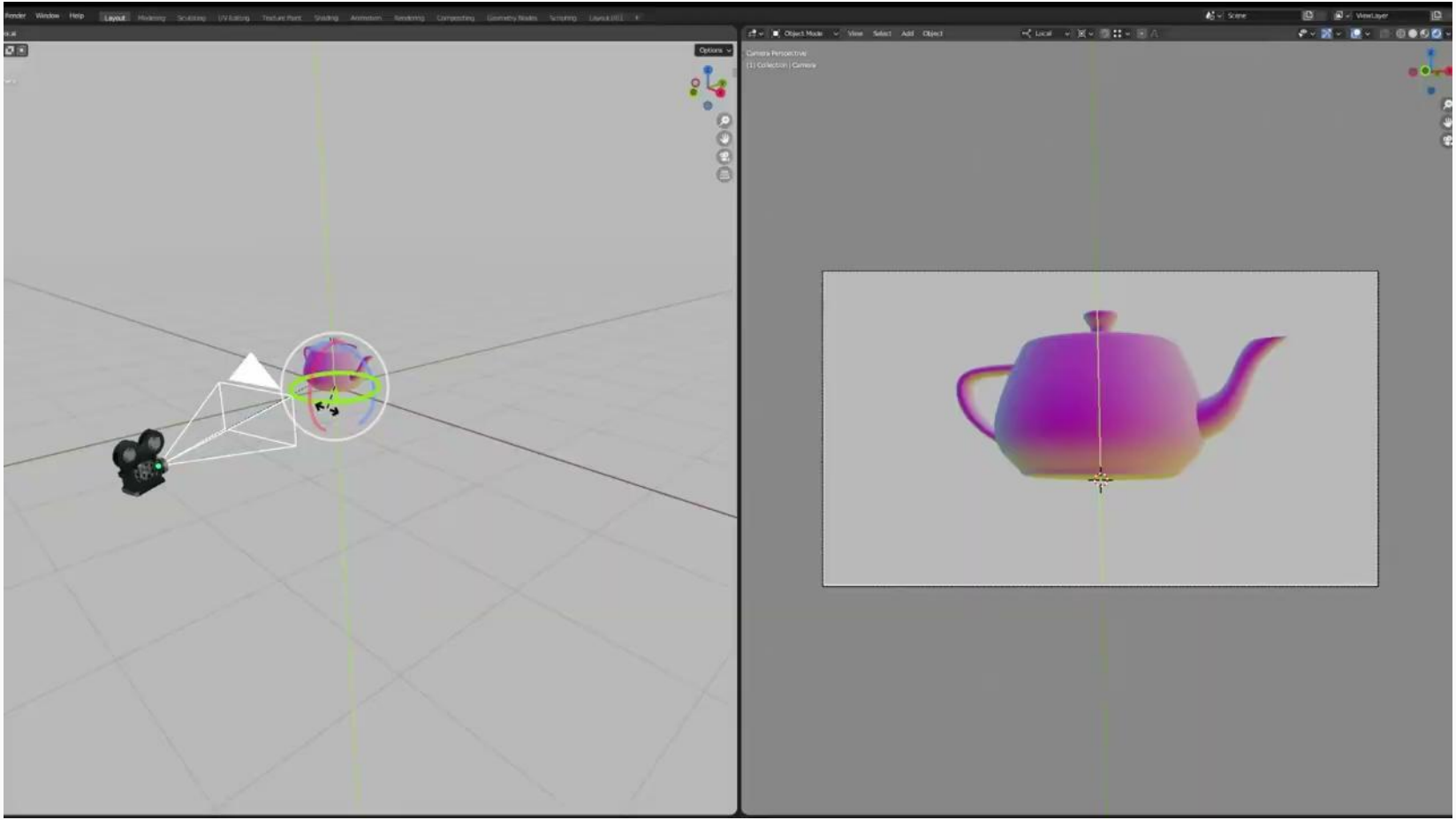
How to Build the View Matrix

$$\vec{c_3} = \text{normalize}(\underline{\text{eye}} - \underline{\text{target}})$$
$$\vec{c_2} = \text{normalize}(\vec{\text{up}} - (\vec{\text{up}} \cdot \vec{c_3})\vec{c_3})$$
$$\vec{c_1} = c_2 \times c_3$$

$$C = \begin{bmatrix} | & | & | & | \\ \vec{c_1} & \vec{c_2} & \vec{c_3} & \underline{\text{eye}} \\ | & | & | & | \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

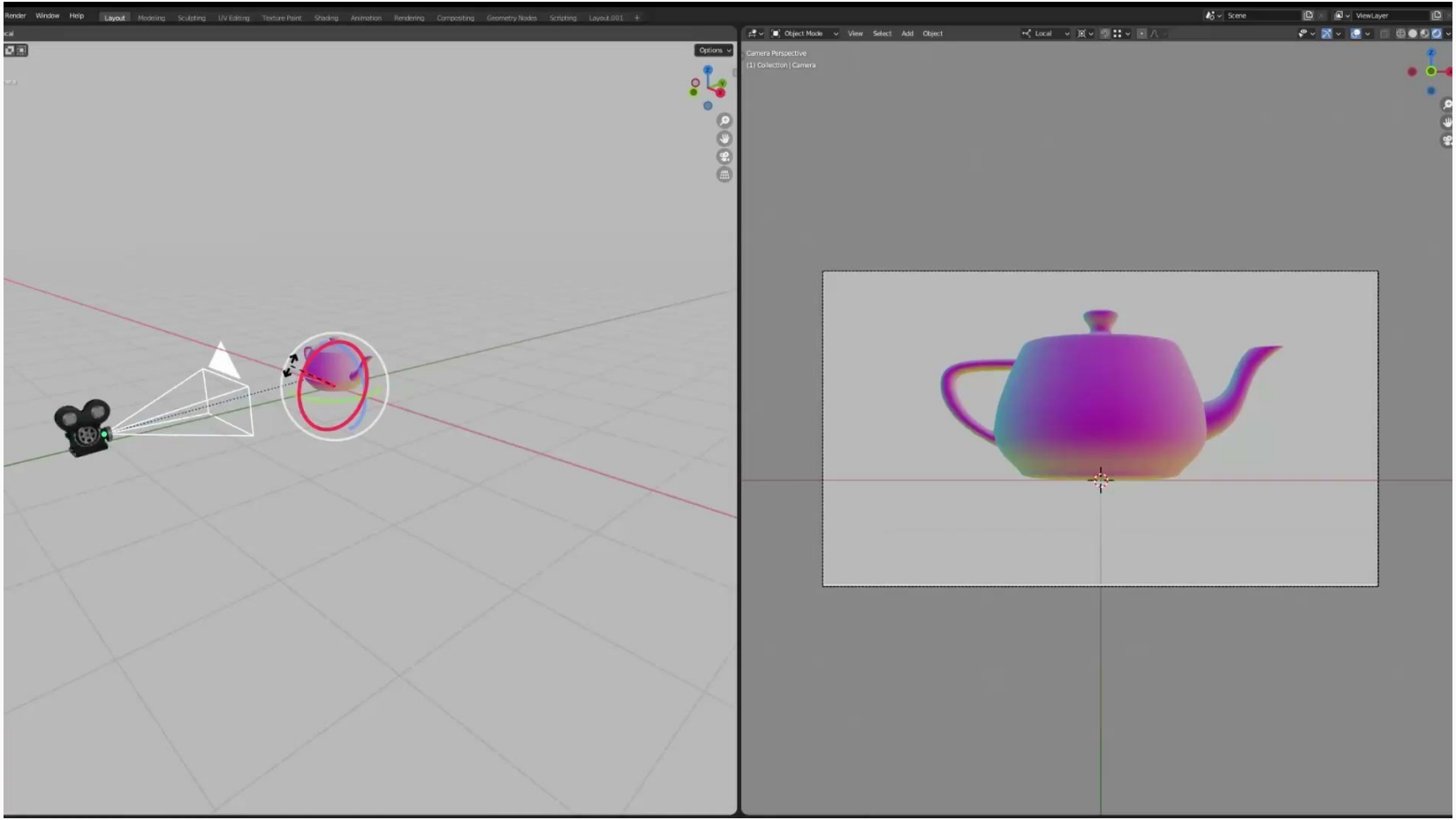


$$V = C^{-1}$$



How to implement `rotateRight()`

Video link: https://drive.google.com/file/d/1JPtCrEwPEeOdUwdAEFn7SKW_BAnMbKDH/view

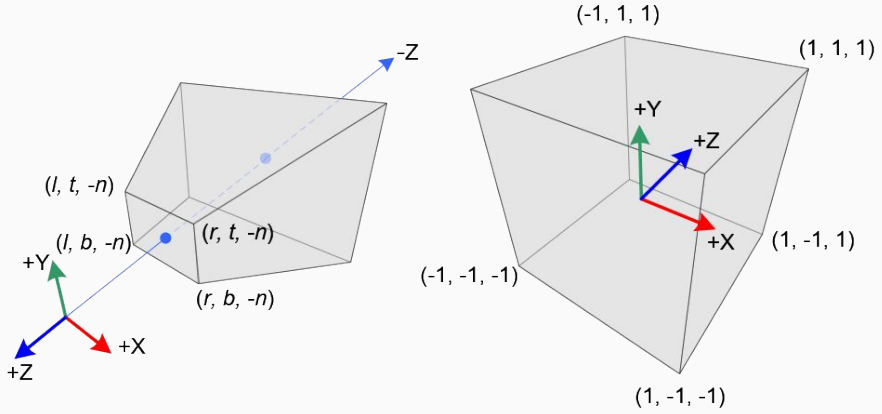


How to implement `rotateUp()`

Video link: https://drive.google.com/file/d/1p6u6geK1a3AhNF8T8OVb8tXNVe4GOf3b/view

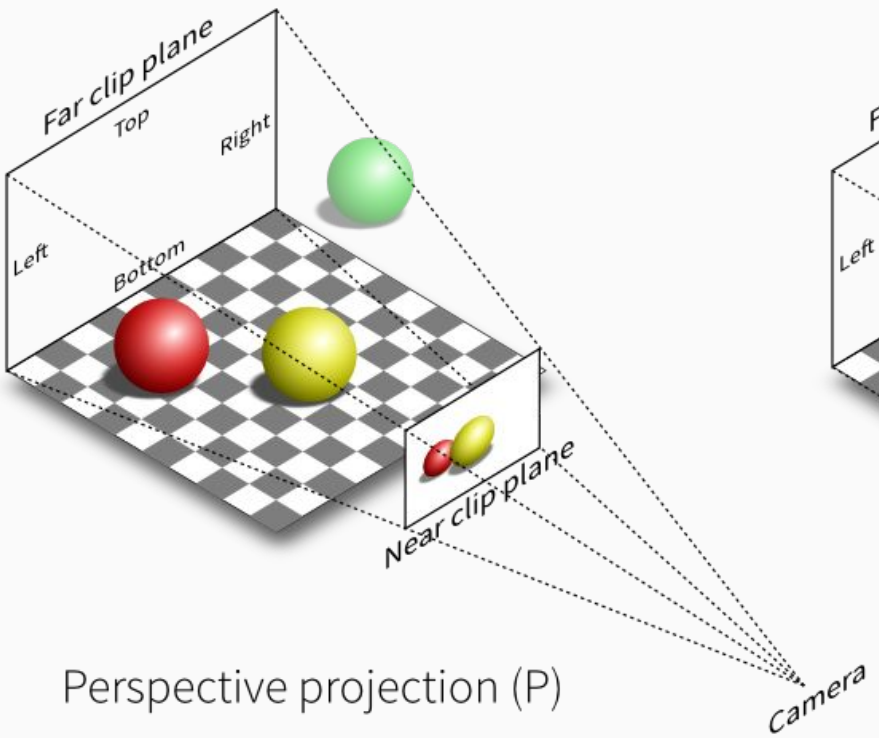
How to Implement `rotateUp/Right()`?

1. Find the camera frame (c1, c2, c3) using `eye`, `target`, and `up`.
2. Find the axis of rotation (hint: ↖ one of these).
3. Rotate the vector that represents the displacement between `eye` and `target`.
4. Update `eye` and `up` correspondingly.

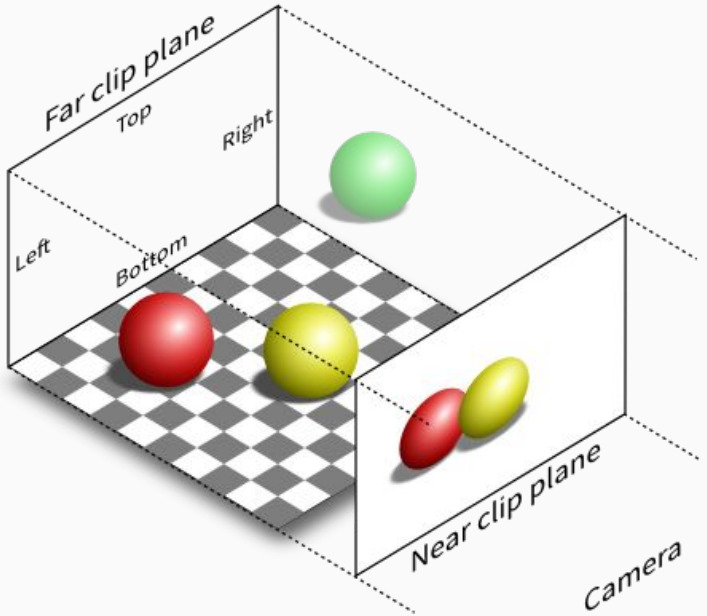


$$P = \begin{bmatrix} \frac{1}{a \tan \theta} & 0 & 0 & 0 \\ 0 & \frac{1}{\tan \theta} & 0 & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

How to build the perspective projection matrix

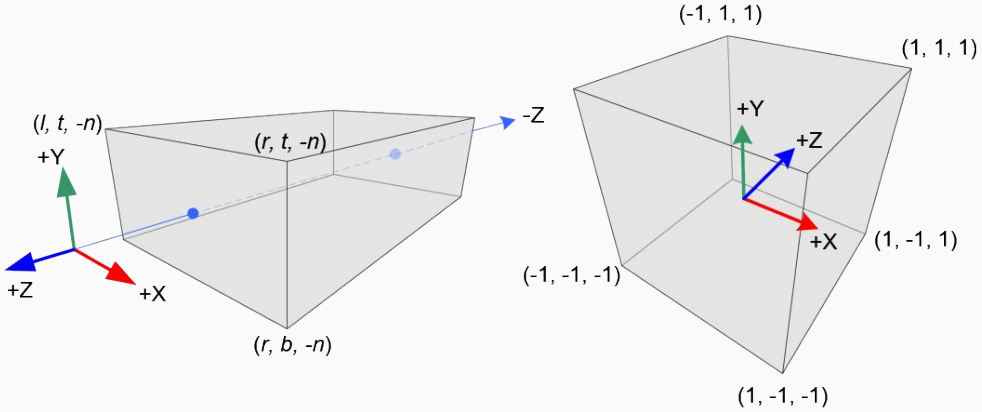


Perspective projection (P)



Orthographic projection (O)

(Extra) A different “perspective” - orthographic projection



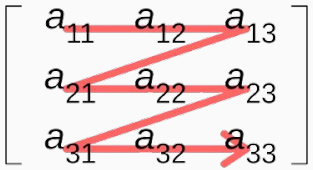
$$O = \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & -\frac{2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(Extra) How to build the orthographic projection matrix

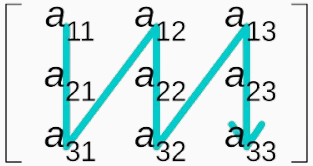
Caveats

1. GLM is ***COLUMN MAJOR***!!!
2. Be mindful of *what* you are rotating.
3. Camera frame should always be orthonormal.

Row-major order



Column-major order



Submission Tips

- Files to submit:
  - `image-00.png, image-01.png, …, image-06.png`
  - `Camera.cpp`
- Autograder:
  - Copy & paste the base64 encoded image to check the difference between yours and the solution. (caveat: due to Gradescope's limitations, the base64 string gets truncated when its too long - i.e. the preview is not available when the difference is too large)