Viewing

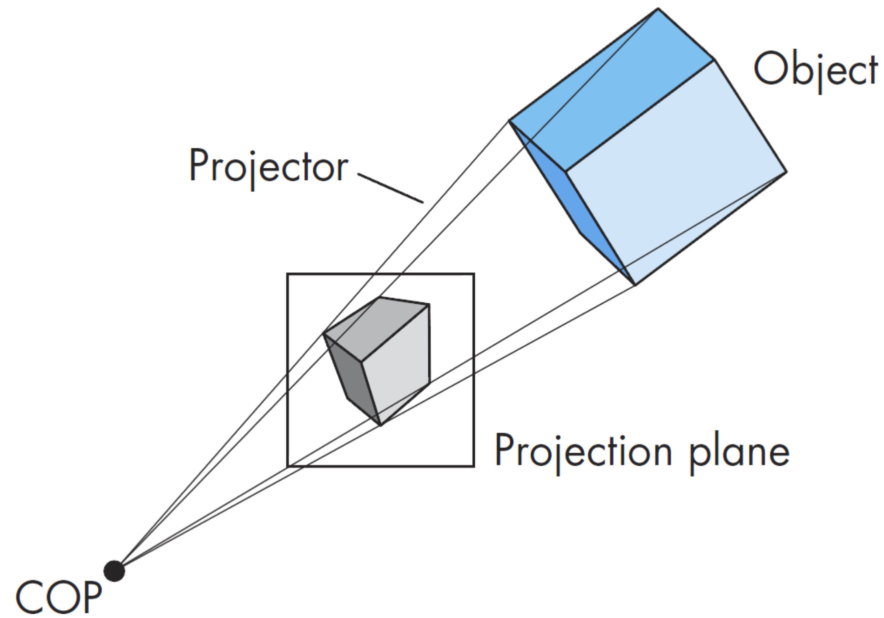
Computer Graphics
Instructor: Sungkil Lee

Today

- **Viewing elements**
- **Perspective and parallel projection**
- **Computer viewing**
  - Forward/Backward approaches
- **Moving Camera: building LookAt matrix**

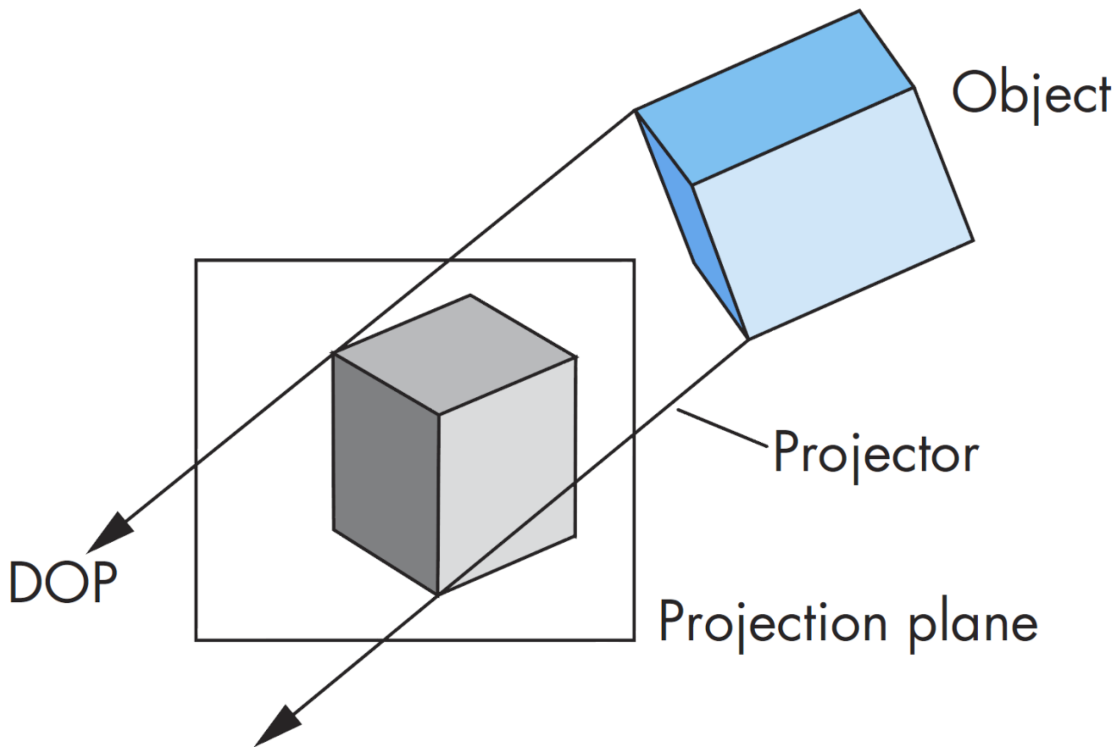
Four Basic Elements of Viewing

- **Objects**
- **Viewer: center of projection (COP)**
  - Corresponds to the center of the lens in the viewer's camera or eyes
  - In CG, COP is the origin of the camera frame for perspective views
- **Projection surface:**
  - Standard projections project onto a ***plane***
- **Projectors**
  - lines that either **converge** at COP or are **parallel**
  - Such projections ***preserve lines***, but not necessarily angles



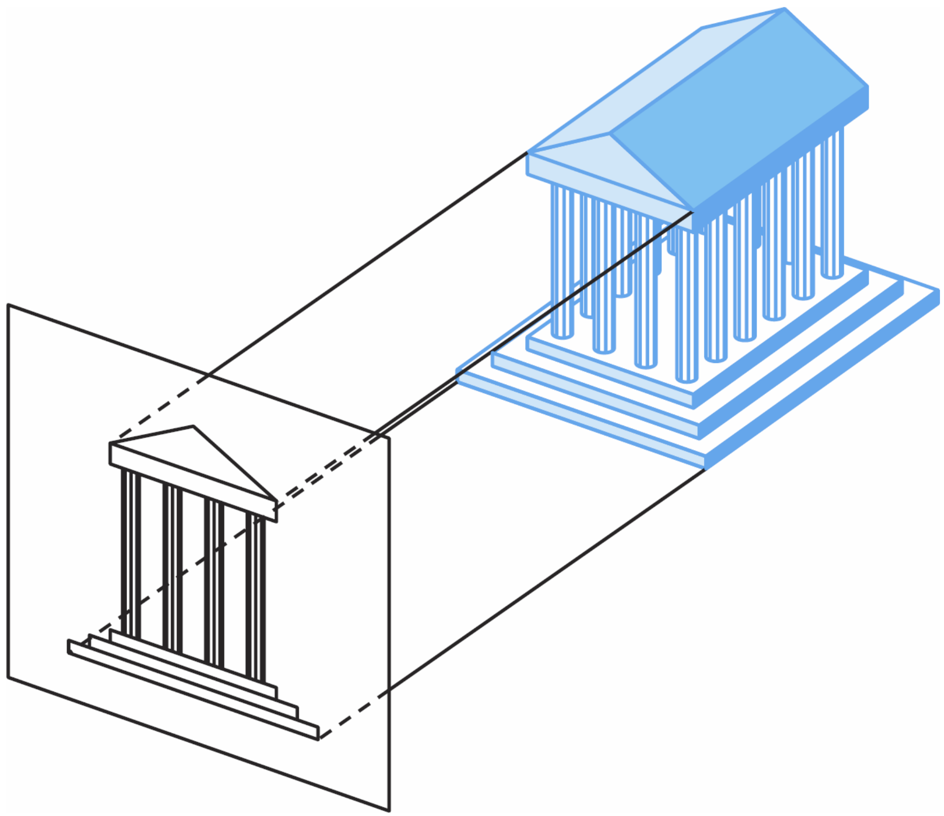
Parallel Projection

- **What if the COP goes to infinity?**
  - The projectors get to be ***parallel***, leading to parallel projection.
  - Then, COP implies no more a point but the direction of projection (***DOP***).



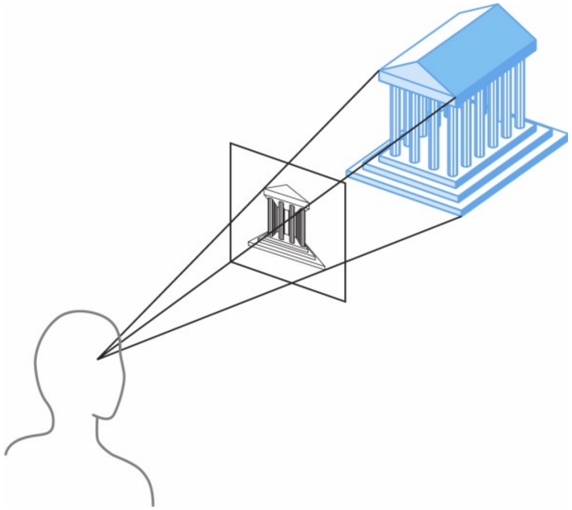
Orthographic Projection

- **A special case of parallel projection**
  - Projectors are ***orthogonal to*** the projection surface.
  - Usually, applied for 2D viewing



Perspective Viewing

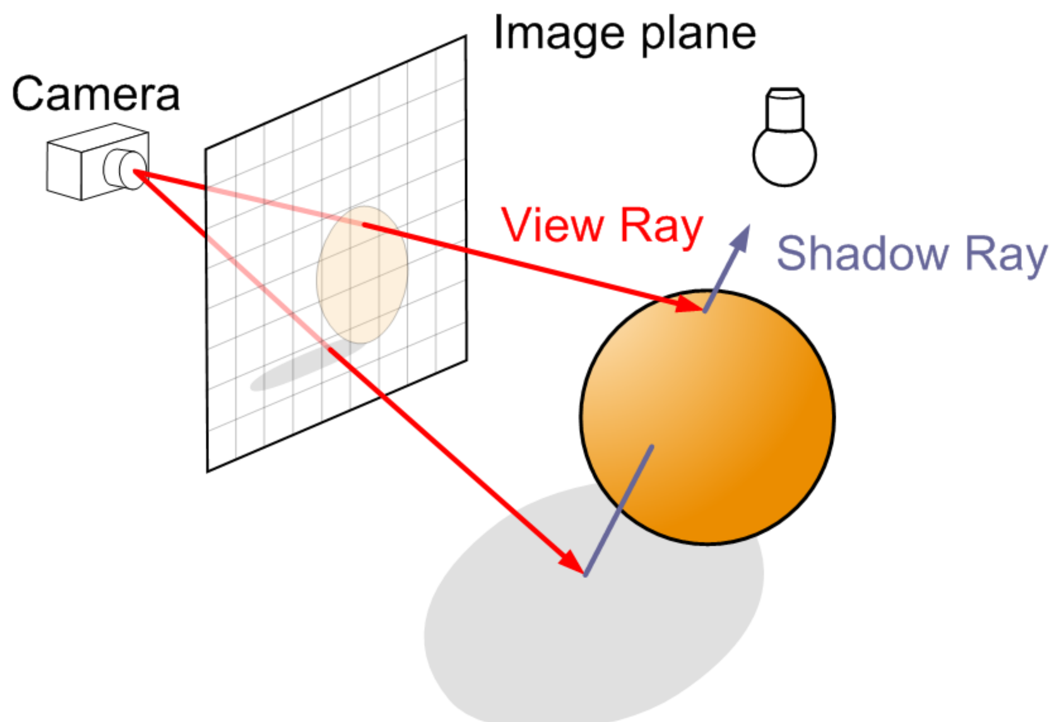
- **Perspective: *diminution***
  - When objects are moved farther from the viewer, their images in a projection surface become ***smaller***.
  - This size change provides natural realism; however, the amount of foreshortening is hard for us to measure.
    - Foreshortening: displaying objects closer (with a shorter depth) due to a different angle of vision (projection).
    - Primary applications are natural-looking images, rather than precise measurements.



Computer Viewing:
Two Viewing Approaches

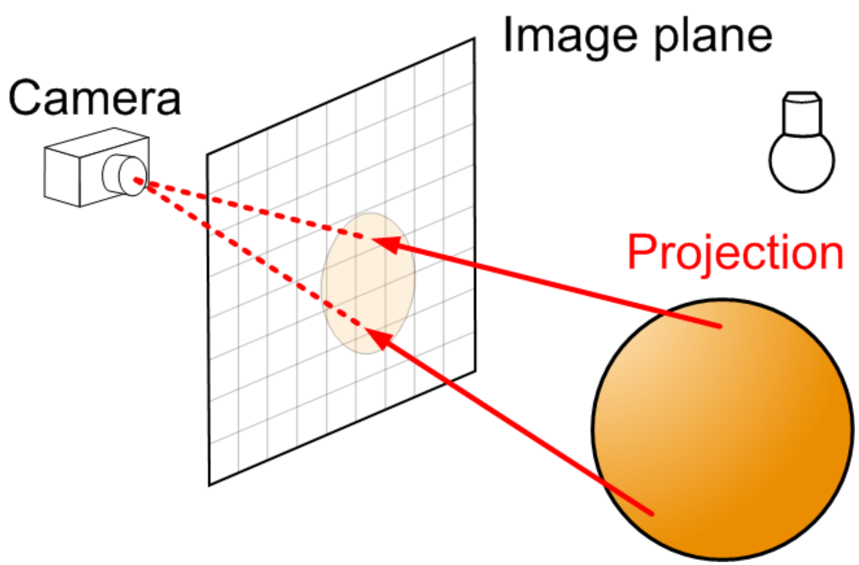
Viewing Approaches

- **Backward approach (ray tracing in CPU)**
  - Start from pixel
  - Explicitly construct the ray corresponding to the pixel
    - The ray that originates in camera, goes through the pixel, and intersects with the surface of some objects.
  - Ask what part of scene intersects with the ray



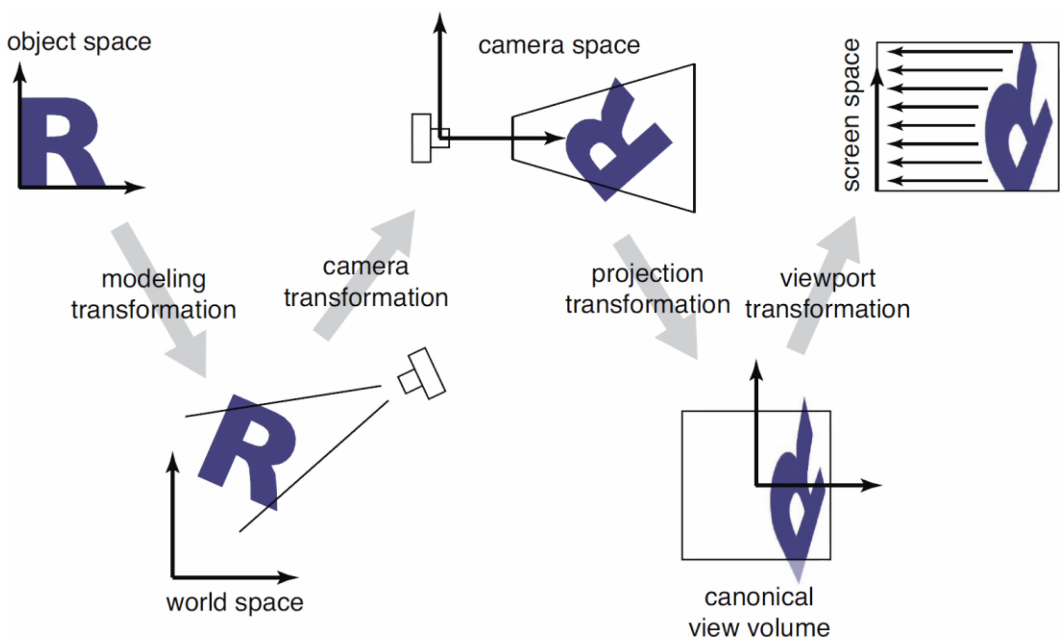
Viewing Approaches

- **Forward approach (pipeline approach)**
  - Naturally, a light ray comes into the image
  - Starting from a point in 3D, computes its projection into the image
  - Central tool is matrix multiplications
    - Combines seamlessly with model and view transformations



Forward Approach: Standard Pipeline

- **Four transformations**
  - Modeling: Local (object) coords. ➔ world coords.
  - View: World coords. ➔ camera coords.
  - Projection: Camera coords. ➔ normalized device coords. (NDC)
  - Viewport: NDC ➔ screen (window) coords.
- **Each transformation corresponds to a matrix multiplication.**
  - Yielding ***a concatenated matrix*** to map a 3D point to its screen position.

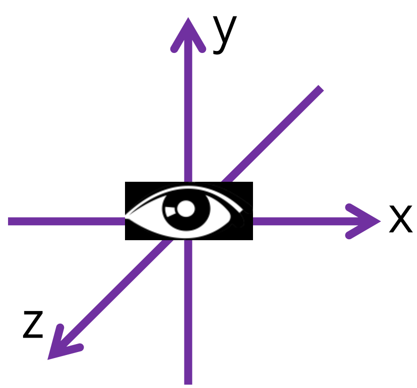


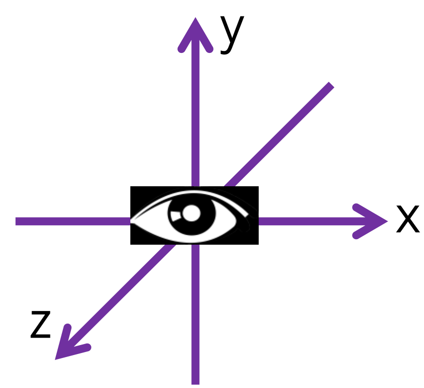
Moving Camera:
Building LookAt Matrix

Model-View Duality

- **If we like to visualize an object, we can either**
  - move camera or move the object in the inverse direction
  - These two actions are actually equivalent (duality).

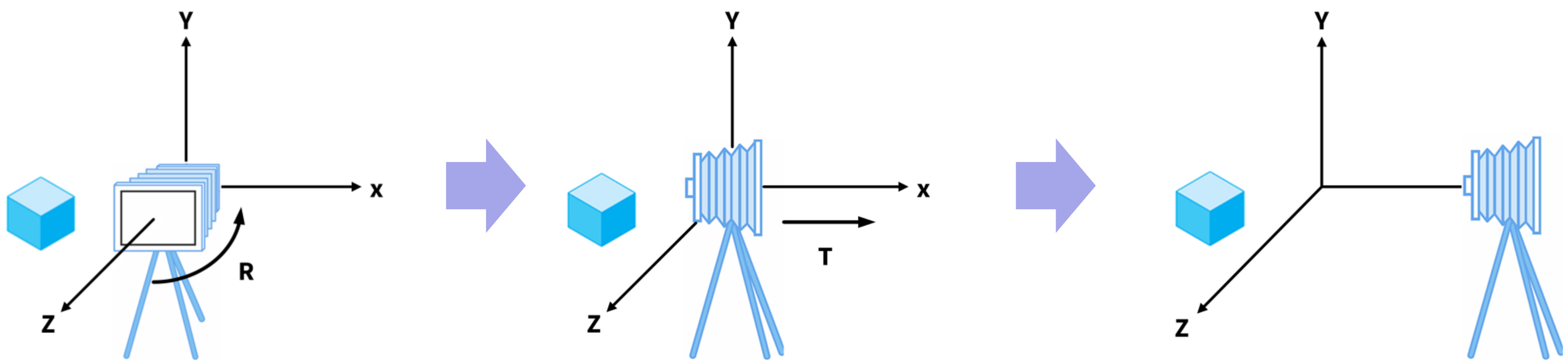
- **Model-view duality**
  - The camera could be understood as being fixed in the origin and the view on the scene is determined by the model-view transformation matrix.





Moving Camera

- **We can move the camera to any desired position by a sequence of rotations and translations**
  - Since this way would be inconvenient, we prefer a more systematic way.

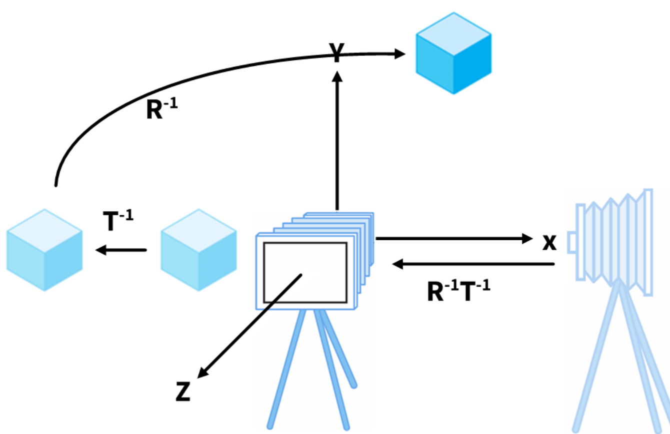


LookAt Method

- **LookAt() method**

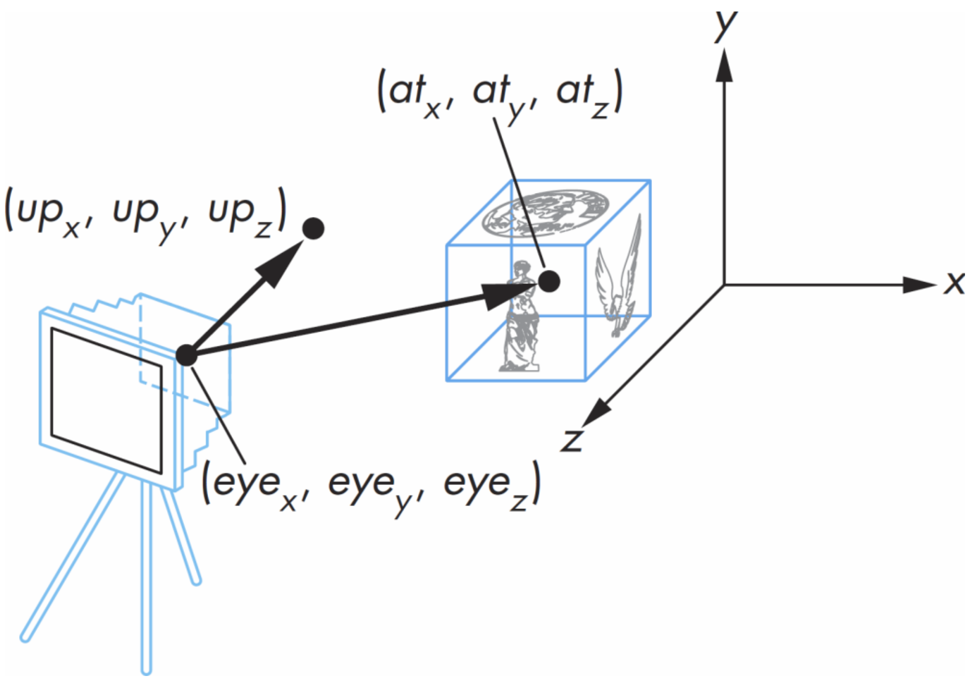
```
mat4 look_at(eye, at, up)
```

  - a more standardized viewing mechanism.
  - **Fix the camera** and move the objects using an inverse matrix multiplication.
  - Hence, we are always in the camera frame (i.e., camera is in the origin).



LookAt Method

- **Viewing specification with (eye, at, up)**
  - eye: a camera's location
  - at: the center of the destination position to be viewed
  - up: upward direction of the camera frame



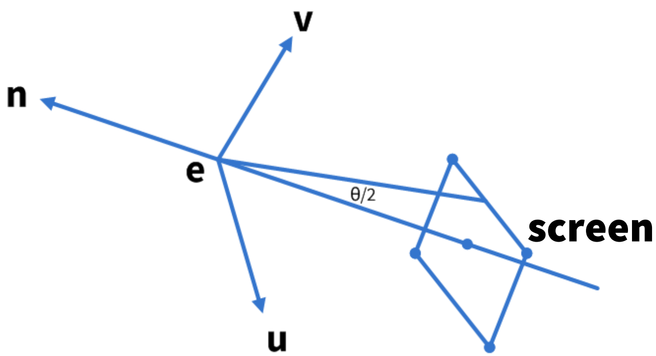
LookAt Matrix

- **eye, at, and up can define a camera *frame*, which has**
  - the origin at eye
  - three basis vectors, $\boldsymbol{u}$, $\boldsymbol{v}$, and $\boldsymbol{n}$ are defined as:

    $\boldsymbol{n}$ = normalize$(\boldsymbol{eye} - \boldsymbol{at})$

    $\boldsymbol{u}$ = normalize$(\boldsymbol{up} \times \boldsymbol{n})$

    $\boldsymbol{v}$ = normalize$(\boldsymbol{n} \times \boldsymbol{u})$
  - They are similar to $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$ in the world space.

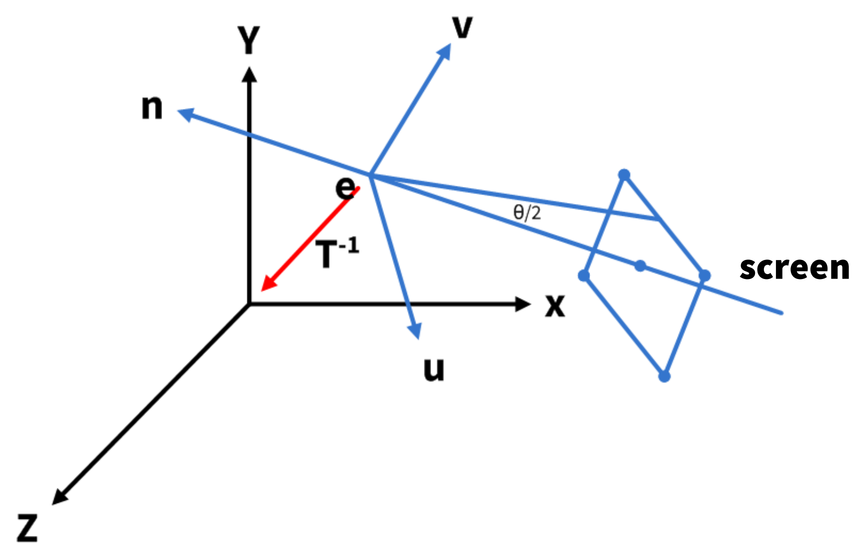


- **Thus, the viewing transformation can be a *change of frame*,**
  - which changes from a world frame to a camera frame.
  - We can do the view transformation with $\mathbf{4 \times 4}$ **LookAt matrix**.

Building LookAt Matrix as Change of Frame

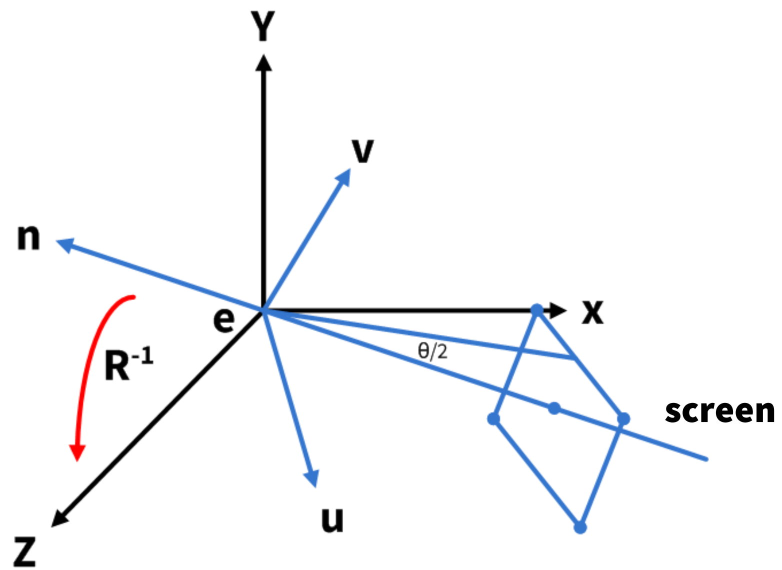
- **First step:**
  - Translation to the negative eye position:

$$\mathbf{T}(-eye) = \begin{bmatrix} 1 & 0 & 0 & -eye.x \\ 0 & 1 & 0 & -eye.y \\ 0 & 0 & 1 & -eye.z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Building LookAt Matrix as Change of Frame

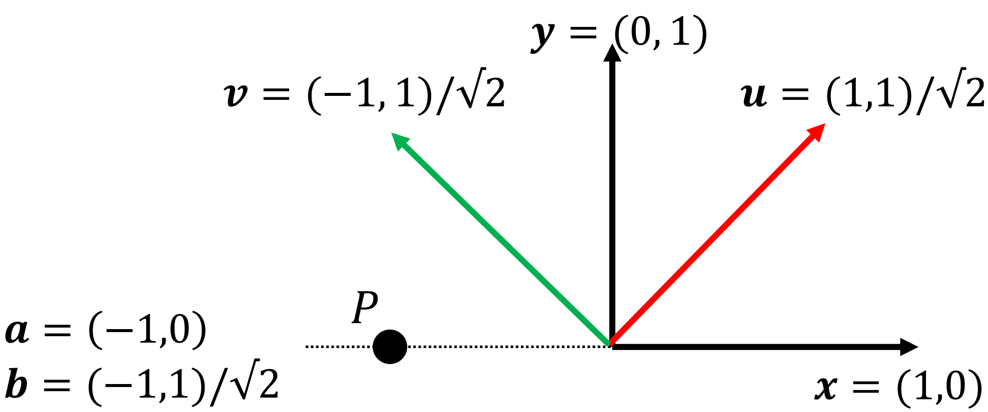
- **Second step: change of coordinate system**
  - Change of **orthonormal basis** = rotation matrix



Building LookAt Matrix as Change of Frame

- **Second step: change of coordinate system**
  - Intuition of the change of coordinate system in 2D
    - A black point P below can be represented in $\boldsymbol{a} = (-1,0)$ and $\boldsymbol{b} = (-1,1)/\sqrt{2}$ with respect to $\{\boldsymbol{x}, \boldsymbol{y}\}$ and $\{\boldsymbol{u}, \boldsymbol{v}\}$, respectively.
    - We can obtain $\boldsymbol{b}$ from $\boldsymbol{a}$ using 2×2 matrix $\mathbf{R}$, which write $(\boldsymbol{u}, \boldsymbol{v})$ in rows:

$$\mathbf{R}\boldsymbol{a} = \begin{bmatrix} u_1 & u_2 \\ v_1 & v_2 \end{bmatrix} \boldsymbol{a} = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix} \begin{bmatrix} -1 \\ 0 \end{bmatrix} = \begin{bmatrix} -1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix} = \boldsymbol{b}$$



Building LookAt Matrix as Change of Frame

- **Second step: change of coordinate system**
  - Given the basis of world coordinate system, $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$, the eye-coordinate basis vectors can be :

$$\boldsymbol{u} = u_1\boldsymbol{x} + u_2\boldsymbol{y} + u_3\boldsymbol{z}$$
$$\boldsymbol{v} = v_1\boldsymbol{x} + v_2\boldsymbol{y} + v_3\boldsymbol{z}$$
$$\boldsymbol{n} = n_1\boldsymbol{x} + n_2\boldsymbol{y} + n_3\boldsymbol{z}$$

  - Then, the relationship can be reformulated as:

$$[\boldsymbol{u} \quad \boldsymbol{v} \quad \boldsymbol{n}] = [\boldsymbol{x} \quad \boldsymbol{y} \quad \boldsymbol{z}]\begin{bmatrix} u_1 & v_1 & n_1 \\ u_2 & v_2 & n_2 \\ u_3 & v_3 & n_3 \end{bmatrix}$$

Building LookAt Matrix as Change of Frame

- **Second step: change of coordinate system**
  - Two representations $\boldsymbol{a}$ and $\boldsymbol{b}$ with respect to $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ and $(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{n})$ can be related as:

$$a_1\boldsymbol{x} + a_2\boldsymbol{y} + a_3\boldsymbol{z} = b_1\boldsymbol{u} + b_2\boldsymbol{v} + b_3\boldsymbol{n}$$

$$\begin{bmatrix} \boldsymbol{x} & \boldsymbol{y} & \boldsymbol{z} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} \boldsymbol{u} & \boldsymbol{v} & \boldsymbol{n} \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} \boldsymbol{x} & \boldsymbol{y} & \boldsymbol{z} \end{bmatrix} \begin{bmatrix} u_1 & v_1 & n_1 \\ u_2 & v_2 & n_2 \\ u_3 & v_3 & n_3 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

  - When we remove $\begin{bmatrix} \boldsymbol{x} & \boldsymbol{y} & \boldsymbol{z} \end{bmatrix}$ from LHS and RHS,

$$\boldsymbol{a} = \begin{bmatrix} u_1 & v_1 & n_1 \\ u_2 & v_2 & n_2 \\ u_3 & v_3 & n_3 \end{bmatrix} \boldsymbol{b}$$

Building LookAt Matrix as Change of Frame

- **Second step: change of coordinate system**

$$\boldsymbol{a} = \mathbf{R}^{\mathrm{T}}\boldsymbol{b}, \quad \text{where} \quad \mathbf{R} = \begin{bmatrix} u_1 & u_2 & u_3 \\ v_1 & v_2 & v_3 \\ n_1 & n_2 & n_3 \end{bmatrix}$$

  - Since the transpose of a matrix of the orthonormal basis (i.e., rotation matrix) is equivalent to the inverse,
$$(\mathbf{R}^{\mathrm{T}})^{-1}\boldsymbol{a} = (\mathbf{R}^{\mathrm{T}})^{\mathrm{T}}\boldsymbol{a} = \mathbf{R}\boldsymbol{a} = \boldsymbol{b}$$
  - Hence, matrix $\mathbf{R}$ transforms the world-coordinate representation $\boldsymbol{a}$ to the eye-coordinate representation $\boldsymbol{b}$.

Building LookAt Matrix as Change of Frame

- **The final** 4×4 **LookAt (view) matrix**
  - By combining the translation and rotation matrices, the final view matrix becomes:

$$\mathbf{RT}(-eye) = \begin{bmatrix} u_1 & u_2 & u_3 & 0 \\ v_1 & v_2 & v_3 & 0 \\ n_1 & n_2 & n_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -eye.x \\ 0 & 1 & 0 & -eye.y \\ 0 & 0 & 1 & -eye.z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Exercise: Change of Frames

- **Do the following on your own.**
  - Derive a matrix that changes a representation of a frame $\{0, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}\}$ to that of a frame $\{0, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{n}\}$. 0 is the origin shared between the two frames.
  - Hint: write $\{\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{n}\}$ in terms of $\{\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}\}$. Then, do as you learn.