Computer Graphics
Camera Control

László Szécsi szecsi@iit.bme.hu

AIT

First Person Camera

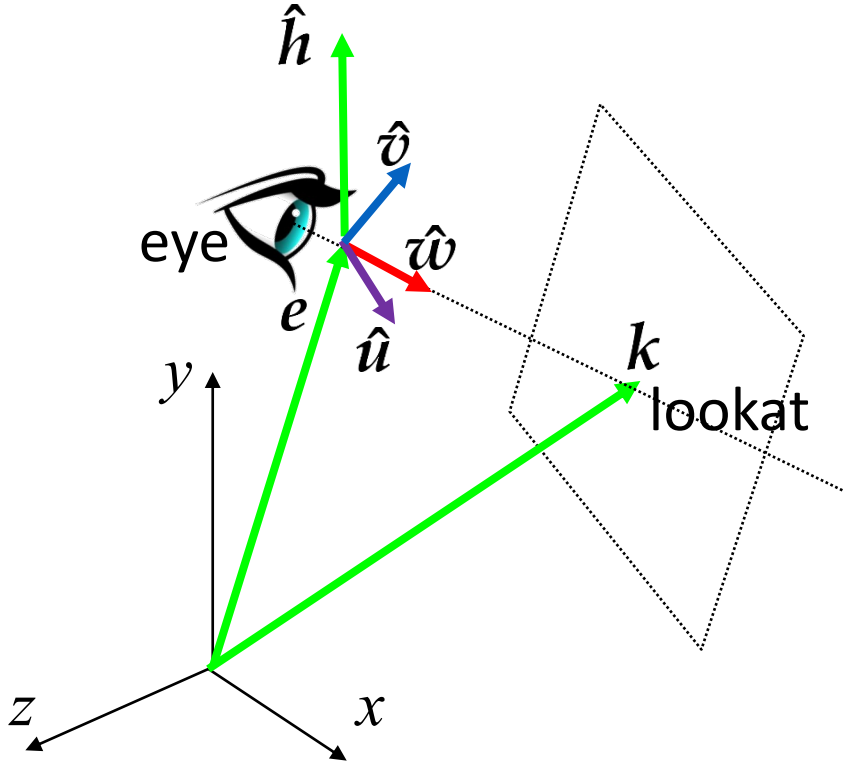
- camera attached to avatar
  - not at the same position (would only see its inside), but moving with it
  - in effect it is like a game object that has the avatar as its parent
  - the camera's view matrix is composed of the avatar's model matrix and its local pose matrix computed with its relative orientation and position (and then inverted)

Third Person Camera

- yaw, pitch, roll unknown
- but we know the look-at point
  - i.e. the avatar's position
- the world-space base vectors of the camera are found using the camera position and look-at point
- the view matrix is assembled directly using these vectors

View transformation: base directions from lookat point

generic upwards



$$\hat{w} = \frac{k - e}{|k - e|} \qquad \hat{u} = \frac{\hat{w} \times \hat{h}}{|\hat{w} \times \hat{h}|} \qquad \hat{v} = \hat{u} \times \hat{w}$$

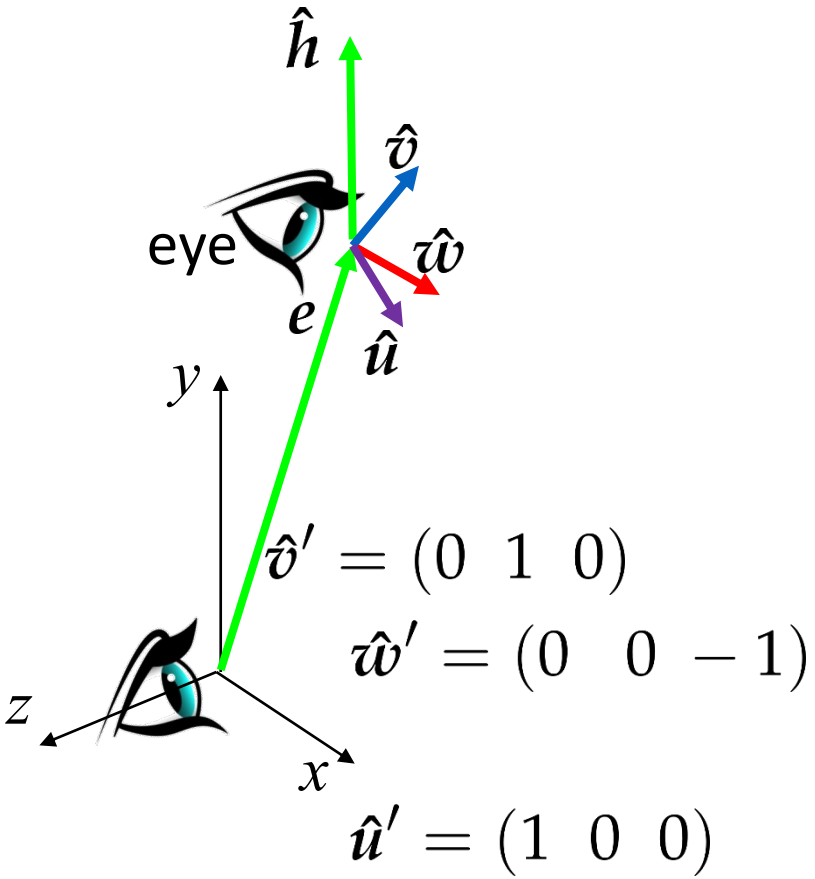
ahead right up

View transformation: matrix from base directions and eye position

$$\hat{w} = \frac{k - e}{|k - e|} \qquad \hat{u} = \frac{\hat{w} \times \hat{h}}{|\hat{w} \times \hat{h}|} \qquad \hat{v} = \hat{u} \times \hat{w}$$

ahead | right | camera up

generic upwards



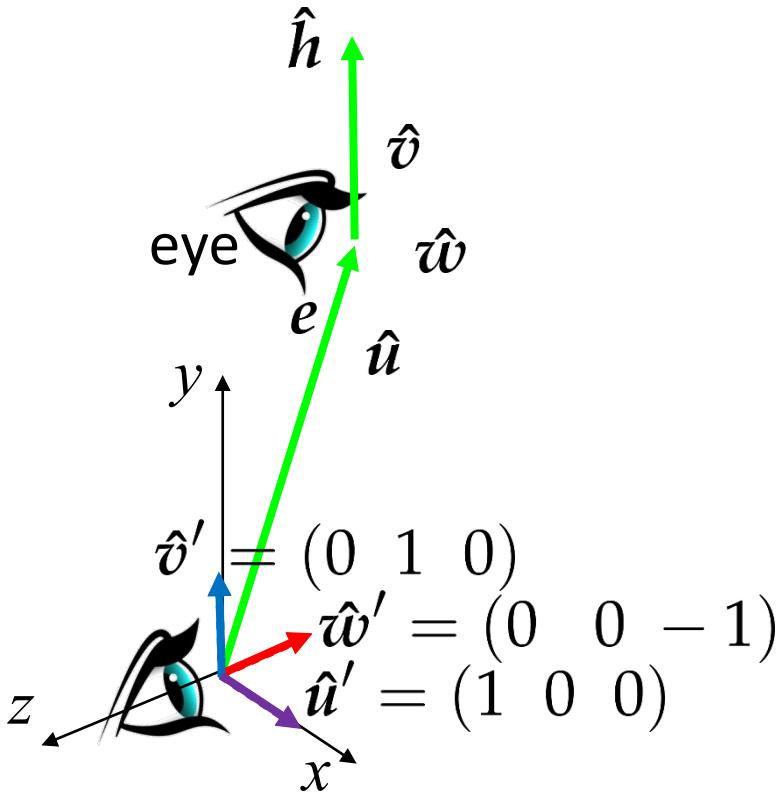
$$\hat{v}' = (0 \;\; 1 \;\; 0)$$

$$\hat{w}' = (0 \;\; 0 \;\; -1)$$

$$\hat{u}' = (1 \;\; 0 \;\; 0)$$

View transformation: matrix from base directions and eye position

generic upwards



$$\hat{w} = \frac{k - e}{|k - e|} \qquad \hat{u} = \frac{\hat{w} \times \hat{h}}{|\hat{w} \times \hat{h}|} \qquad \hat{v} = \hat{u} \times \hat{w}$$

ahead right camera up

$$V = \left(\begin{bmatrix} u_x & u_y & u_z & 0 \\ v_x & v_y & v_z & 0 \\ -w_x & -w_y & -w_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ e_x & e_y & e_z & 1 \end{bmatrix} \right)^{-1}$$

$$\breve{r}_{\text{camera}} = \breve{r}_{\text{world}} V$$

Generic up direction in the world

```
PerspectiveCamera.worldUp = new Vec3(0, 1, 0);
```

Base directions

```
this.right.setVectorProduct(
   this.ahead,
   PerspectiveCamera.worldUp );
this.right.normalize();
this.up.setVectorProduct(this.right, this.ahead);
```

Compute view matrix

```
this.viewMatrix.set(
  this.right.x   ,  this.right.y ,  this.right.z   , 0,
  this.up.x      ,  this.up.y    ,  this.up.z      , 0,
 -this.ahead.x   , -this.ahead.y , -this.ahead.z   , 0,
  0              ,  0             , 0               , 1).
                                translate(this.position).
                                invert();

this.viewProjMatrix.set(this.viewMatrix).mul(this.projMatrix);
```