Moving the Camera

Lecture 13

Robb T. Koether

Hampden-Sydney College

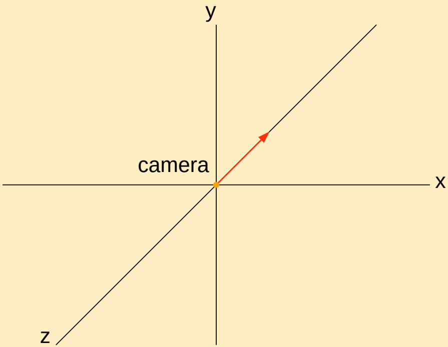
Mon, Sep 23, 2019

Outline

Outline

The Viewing Transformation

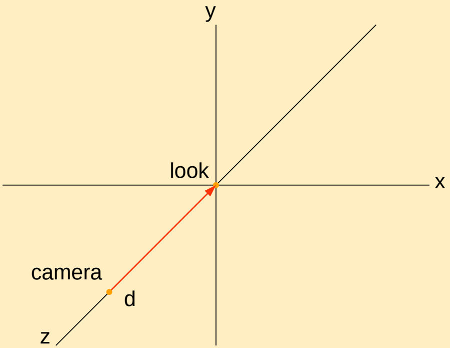
The Viewing Transformation



The default camera

The Viewing Transformation

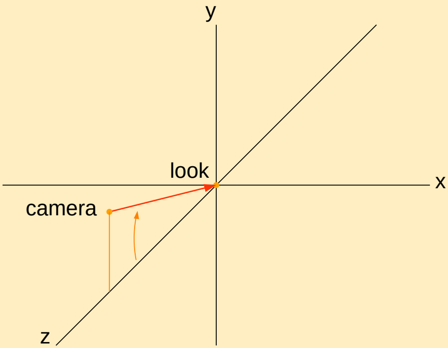
The Viewing Transformation



Translate the camera

The Viewing Transformation

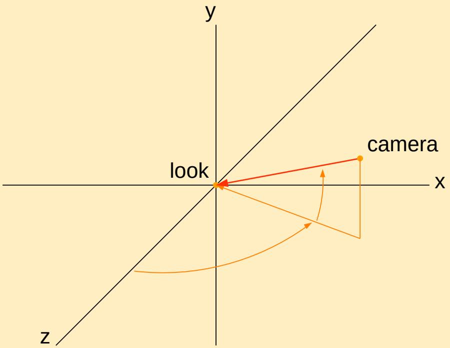
The Viewing Transformation



Rotate the camera vertically (pitch)

The Viewing Transformation

The Viewing Transformation



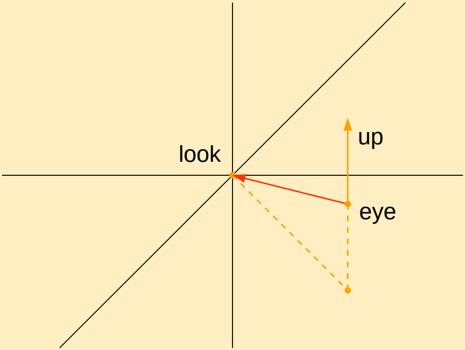
Rotate the camera horizontally (yaw)

The Camera's Position

- The camera's position may be determined by three quantities.
  - Pitch – angle tilting forward (up or down).
  - Yaw – angle left or right.
  - Distance – distance from the look point.
- Given *pitch*, *yaw*, and *dist*, how do we compute the x-, y-, and z-coordinates of the camera?

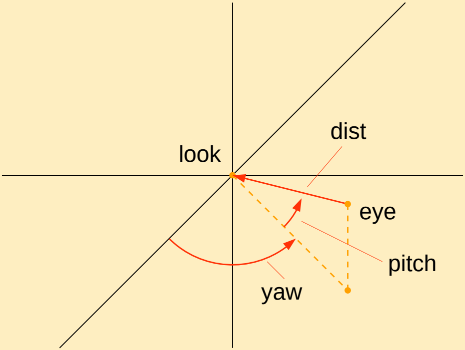
The Camera's Position

The Camera's Position



The *eye* (or camera) position, the *look* point, and *up*

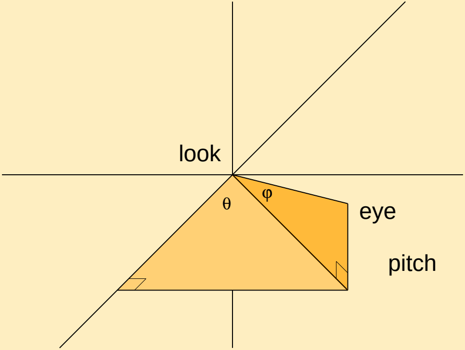
The Camera's Position



The *pitch*, *yaw*, and *dist*

The Camera's Position

The Camera's Position



Let φ be the pitch and θ the yaw

Outline

Calculating the Camera's Coordinates

- The vertical distance (elevation, or *y*) from the *xz*-plane to *eye* is

$$dist \cdot \sin\varphi.$$

Calculating the Camera's Coordinates

- The vertical distance (elevation, or *y*) from the *xz*-plane to *eye* is

$$\mathit{dist} \cdot \sin \varphi.$$

- The horizontal distance from *look* to directly under *eye* is

$$\mathit{dist} \cdot \cos \varphi.$$

Calculating the Camera's Coordinates

- The vertical distance (elevation, or *y*) from the *xz*-plane to *eye* is

$$\mathit{dist} \cdot \sin \varphi.$$

- The horizontal distance from *look* to directly under *eye* is

$$\mathit{dist} \cdot \cos \varphi.$$

- Thus, the *x* coordinate is

$$(\mathit{dist} \cdot \cos \varphi) \sin \theta$$

  and the *z*-coordinate is

$$(\mathit{dist} \cdot \cos \varphi) \cos \theta.$$

Calculating the Camera's Coordinates

Calculating the Camera's Coordinates

```
eye = dist*vec3(cos(pitch)*sin(yaw), sin(pitch),
        cos(pitch)*cos(yaw));
```

$$x = \mathit{dist} \cdot \cos\varphi \sin\theta,$$
$$y = \mathit{dist} \cdot \sin\varphi,$$
$$z = \mathit{dist} \cdot \cos\varphi \cos\theta.$$

- This calculation of `eye` should be placed in the `setView()` function.

Outline

Moving the Camera

- To move the camera, we will modify *yaw*, *pitch*, and *dist*.
- We want the user interface to be simple and intuitive.
  - Drag the mouse left or right to change *yaw*.
  - Drag the mouse up or down to change *pitch*.
  - Roll the mouse wheel to change *dist*.

The Yaw Angle

- For the yaw angle, let the width of the window represent 180°.
- Let `old_x` and `old_y` be where the mouse was last clicked, as reported by the mousebutton callback function (and later updated in the cursor position callback function).
- In the cursor position callback function, `xpos` and `ypos` will be the current coordinates.

The Yaw Angle

The Yaw Angle

```
float d_yaw = (float)(xpos - old_x)/fb_width*180.0f;
yaw += d_yaw;
old_x = xpos;
```

- Write similar code for `pitch`.

The Distance

- For the distance to the look point, let each click of the wheel represent a 5% change.
- The change should be small enough that zooming appears to be smooth.
- A forward rotation will replace `dist` with `dist/1.05f`.
- A backward rotation will replace `dist` with `1.05f*dist`.

The Distance

The Distance

```
if (yoffset > 0)
    dist /= 1.05f;
else
    dist *= 1.05f;
```

Outline

Assignment

Assignment

- Assignment 12.