The Projection Matrix

Lecture 25

Robb T. Koether

Hampden-Sydney College

Wed, Nov 6, 2019

Outline

Outline

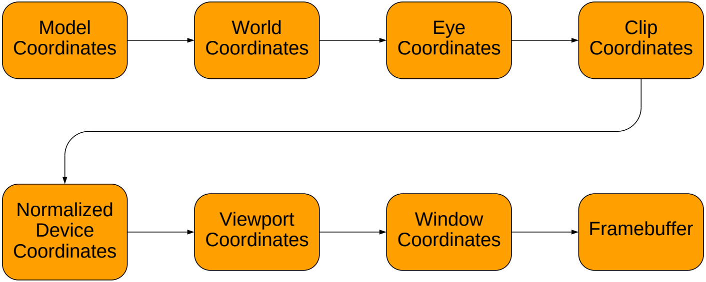
Debugging Tip of the Day

Debugging Tip of the Day

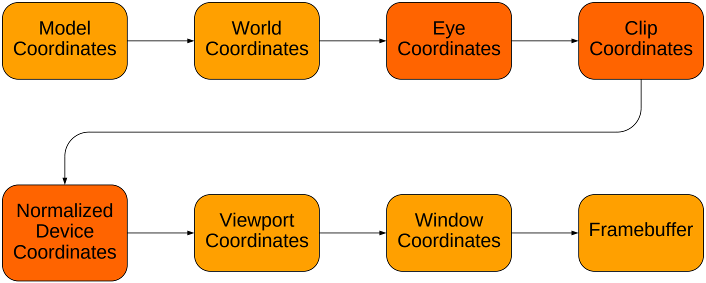
- To locate the statement causes the program to crash, first comment out all statements within the function.
- Run the program.
- Then uncomment the statements one by one, running the program each time until it crashes.
- At that point, you have found the statement that is causing the crash.

Outline

The Graphics Pipeline



The Graphics Pipeline



Homogeneous Coordinates

- Points are stored in homogeneous coordinates (x, y, z, w).
- The true 3D coordinates are $\left(\frac{x}{w}, \frac{y}{w}, \frac{z}{w}\right)$.
- Therefore, for example, the points $(4, 3, 2, 1)$ and $(8, 6, 4, 2)$ represent the same 3D point $(4, 3, 2)$.
- This fact will play a crucial role in the projection matrix.

Coordinate Systems

- Eye coordinates
  - The camera is at the origin, looking in the negative z-direction.
  - View frustrum (right, left, bottom, top, near, far).
- Normalized device coordinates

$$-1 \leq x \leq 1$$
$$-1 \leq y \leq 1$$
$$-1 \leq z \leq 1$$

Outline

The Transformation

- Points in eye coordinates must be transformed into normalized device coordinates.
- But first they are transformed to clipping coordinates.

The Transformation

- For example, the near-upper-right corner $(r, t, -n, 1)$ in eye coordinates is transformed to $(n, n, -n, n)$ in clip coordinates.
- The far-bottom-left corner $\left(l\left(\frac{f}{n}\right), b\left(\frac{f}{n}\right), -f, 1\right)$ in eye coordinates is transformed to $(-f, -f, f, f)$ in clip coordinates.
- This is done in two steps.

The Transformation

- For example, the near-upper-right corner $(r, t, -n, 1)$ in eye coordinates is transformed to $(n, n, -n, n)$ in clip coordinates.
- The far-bottom-left corner $\left(l\left(\frac{f}{n}\right), b\left(\frac{f}{n}\right), -f, 1\right)$ in eye coordinates is transformed to $(-f, -f, f, f)$ in clip coordinates.
- This is done in two steps.
- By the way, this is why the ratio $\frac{f}{n}$ should not be too large.

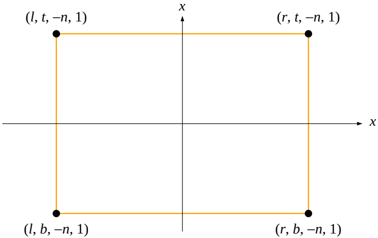
The Transformation

- For example, the near-upper-right corner $(r, t, -n, 1)$ in eye coordinates is transformed to $(n, n, -n, n)$ in clip coordinates.
- The far-bottom-left corner $\left(l\left(\frac{f}{n}\right), b\left(\frac{f}{n}\right), -f, 1\right)$ in eye coordinates is transformed to $(-f, -f, f, f)$ in clip coordinates.
- This is done in two steps.
- By the way, this is why the ratio $\frac{f}{n}$ should not be too large.
- Erik, what happens if $\frac{f}{n}$ is too large?

The Perspective Transformation

- In the first step (near plane),

$$\begin{aligned}
(r, t, -n, 1) &\rightarrow (nr, nt, -n, n)\\
(l, t, -n, 1) &\rightarrow (nl, nt, -n, n)\\
(r, b, -n, 1) &\rightarrow (nr, nb, -n, n)\\
(l, b, -n, 1) &\rightarrow (nl, nb, -n, n)
\end{aligned}$$

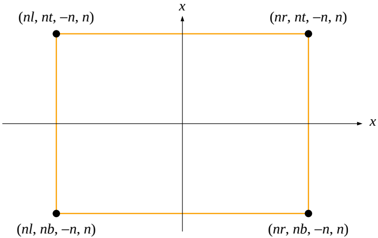


The near plane

The Perspective Transformation

- In the first step (near plane),

$$\begin{aligned}(r, t, -n, 1) &\rightarrow (nr, nt, -n, n)\\(l, t, -n, 1) &\rightarrow (nl, nt, -n, n)\\(r, b, -n, 1) &\rightarrow (nr, nb, -n, n)\\(l, b, -n, 1) &\rightarrow (nl, nb, -n, n)\end{aligned}$$



The near plane

The Perspective Transformation

- and

$$\left(r\left(\frac{f}{n}\right), t\left(\frac{f}{n}\right), -f, 1\right) \rightarrow (fr, ft, f, f)$$
$$\left(l\left(\frac{f}{n}\right), t\left(\frac{f}{n}\right), -f, 1\right) \rightarrow (fl, ft, f, f)$$
$$\left(r\left(\frac{f}{n}\right), b\left(\frac{f}{n}\right), -f, 1\right) \rightarrow (fr, fb, f, f)$$
$$\left(l\left(\frac{f}{n}\right), b\left(\frac{f}{n}\right), -f, 1\right) \rightarrow (fl, fb, f, f).$$

The Perspective Transformation

- This is accomplished by the perspective matrix is

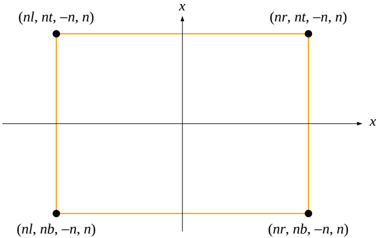
$$\mathbf{P}_1 = \begin{pmatrix} n & 0 & 0 & 0 \\ 0 & n & 0 & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix}.$$

- Note the bottom row.

The Perspective Transformation

- In the second step,

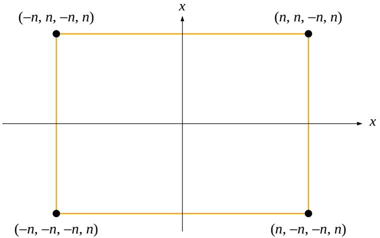
$$
\begin{aligned}
(nr, nt, -n, n) &\rightarrow (n, n, -n, n) \\
(nl, nt, -n, n) &\rightarrow (-n, n, -n, n) \\
(nr, nb, -n, n) &\rightarrow (n, -n, -n, n) \\
(nl, nb, -n, n) &\rightarrow (-n, n, -n, n)
\end{aligned}
$$



The near plane

- In the second step,

$$
\begin{aligned}
(nr, nt, -n, n) &\rightarrow (n, n, -n, n) \\
(nl, nt, -n, n) &\rightarrow (-n, n, -n, n) \\
(nr, nb, -n, n) &\rightarrow (n, -n, -n, n) \\
(nl, nb, -n, n) &\rightarrow (-n, n, -n, n)
\end{aligned}
$$



The near plane

The Perspective Transformation

- and

$$\begin{aligned}(fr, ft, f, f) &\rightarrow (f, f, f, f)\\(fl, ft, f, f) &\rightarrow (-f, f, f, f)\\(fr, fb, f, f) &\rightarrow (f, -f, f, f)\\(fl, fb, f, f) &\rightarrow (-f, -f, f, f).\end{aligned}$$

The Projection Transformation

- This is accomplished by the matrix

$$\mathbf{P}_2 = \begin{pmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

The Projection Matrix

- The product of the two transformations is the projection matrix.
- It is the matrix that transforms points from eye coordinates to clip coordinates.

$$\mathbf{P} = \mathbf{P}_2\mathbf{P}_1 = \begin{pmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix}.$$

Clipping Coordinates

- In clip coordinates, a point $P(x, y, z, w)$ is clipped if

$$|x| > w \text{ or } |y| > w \text{ or } |z| > w.$$

Outline

The Transformation

- This is followed by the homogeneous divide, or perspective division.
- It is a *nonlinear* transformation.
- It transforms clip coordinates to normalized device coordinates.
- For example,

$$
\begin{aligned}
(n, n, -n, n) &\rightarrow (1, 1, -1) \\
(-f, -f, f, f) &\rightarrow (-1, -1, 1)
\end{aligned}
$$

Outline

The Projection Matrix

Example (Creating the Projection Matrix)

```
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
glFrustum(l, r, b, t, n, f);
```

- The function

  ```
  glFrustum(l, r, b, t, n, f)
  ```

  creates this matrix and multiplies the current projection matrix by it.

The Projection Matrix

- The function

  `gluPerspective(`*`angle`*`, `*`ratio`*`, `*`near`*`, `*`far`*`)`

  also creates the projection matrix by calculating r, l, t, and b.

The Projection Matrix

- The formulas are

$$\begin{aligned} t &= n\tan\left(\frac{\texttt{angle}}{2}\right) \\ b &= -t \\ r &= t\cdot\texttt{ratio} \\ l &= -r \\ n &= \texttt{near} \\ f &= \texttt{far} \end{aligned}$$

Question

- When choosing the near and far planes in the `gluPerspective()` call, why not let n be very small, say 0.000001, and let f be very large, say 1000000.0?

Outline

Orthogonal Projections

- The matrix for an orthogonal projection is much simpler.
- All it does is rescale the x-, y-, and z-coordinates to $[-1, 1]$.
- The positive direction of z is reversed.
- It represents a linear transformation; the w-coordinate remains 1.

Orthogonal Projections

- The matrix of an orthogonal projection is

$$\mathbf{P} = \begin{pmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & -\frac{2}{f-n} & \frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Outline

Homework

Homework

- Read Section 4.4 – Parallel projections.
- Read Section 4.5 – Perspective projections.
- Read Section 4.6 – Perspective projections in OpenGL.
- Read Section 4.7 – Perspective-projection matrices.