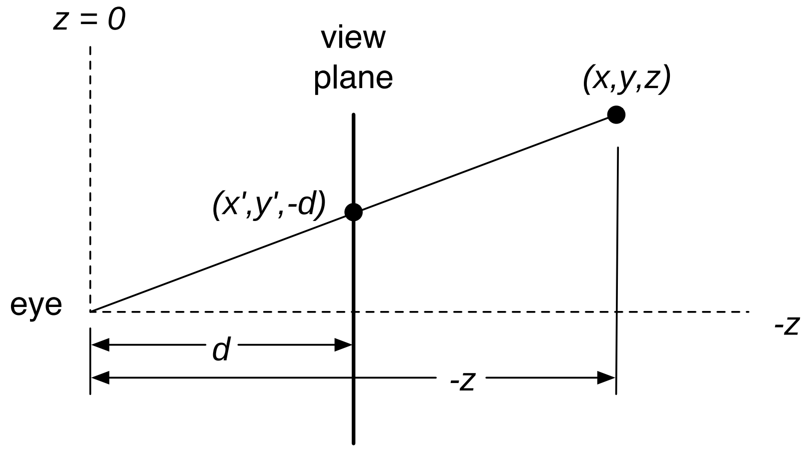
Perspective Projections

CS 442/452

September 17, 2008

Perspective Projection onto View Plane



Via similar triangles (note $z < 0$):

$$\begin{aligned} \frac{x'}{d} &= \frac{x}{-z} \\ \frac{y'}{d} &= \frac{y}{-z} \\ z' &= -d \end{aligned}$$

$$(x', y', z') = \left(\frac{xd}{-z}, \frac{yd}{-z}, -d \right).$$

Perspective Transformation

- Our perspective transformation is <u>not</u> linear (we can <u>not</u> describe it with a (simple) matrix).
- We exploit the use of homogeneous coordinates

$$(x, y, z, w) \equiv (x/w, y/w, z/w, 1) \qquad (w \neq 0).$$

- We then create the following perspective matrix

$$\begin{bmatrix} xd \\ yd \\ zd \\ -z \end{bmatrix} = \begin{bmatrix} d & 0 & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & d & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

  After transforming our point, we perform a homogeneous **perspective division**:

$$(xd, yd, zd, -z) \mapsto \left(\frac{-xd}{z}, \frac{-yd}{z}, -d, 1 \right)$$

Loss of Depth Information

$$\begin{bmatrix} d & 0 & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & d & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} xd \\ yd \\ zd \\ -z \end{bmatrix} \mapsto \begin{bmatrix} \frac{-xd}{z} \\ \frac{-yd}{z} \\ -d \\ 1 \end{bmatrix}$$

- Our perspective matrix is (unfortunately) singular (obvious from column of 0's).
- Matrix is not invertible (3-D objects flatted to 2-D).
- Depth information is lost (we need depth information for visibility sorting).

An Invertible Perspective Matrix

We alter our perspective transformation by choosing some non-zero values for a and b and defining the following transformation:

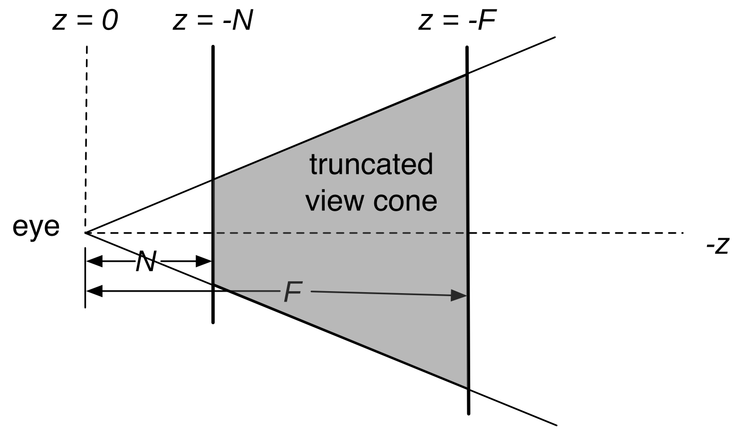
$$\begin{bmatrix} d & 0 & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & a & b \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} xd \\ yd \\ az + b \\ -z \end{bmatrix} \mapsto \begin{bmatrix} \frac{-xd}{z} \\ \frac{-yd}{z} \\ -a - \frac{b}{z} \\ 1 \end{bmatrix}$$

Relative depth information will be preserved in the third component as a (non-linear) function f of the true depth z :

$$f(z) = -a - \frac{b}{z}.$$

Near and Far Clipping Planes

The user specifies the distance $N > 0$ to the **near** clipping plane and the distance $F > N$ to the **far** clipping plane.

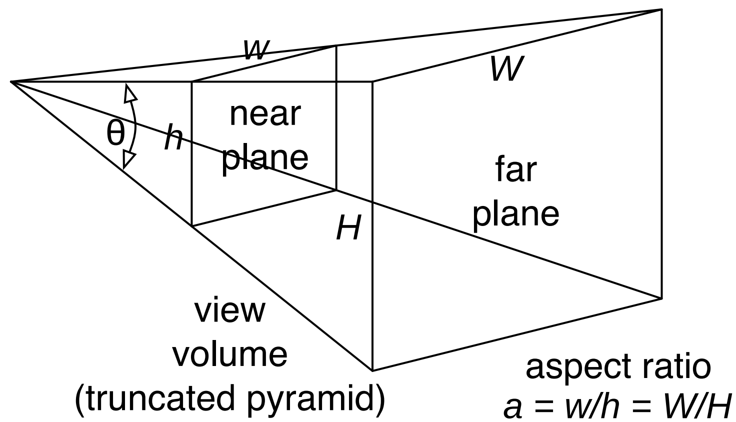


We choose a and b so that the near and far planes are mapped to -1 and $+1$ respectively:

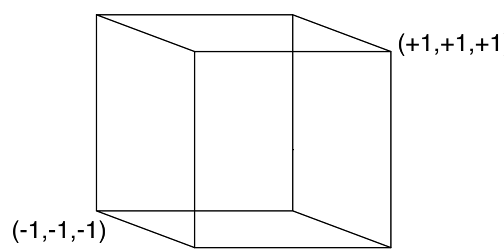
$$f(-N) = -a + \frac{b}{N} \quad = \quad -1$$
$$f(-F) = -a + \frac{b}{F} \quad = \quad +1.$$

Thus we get $a = -\frac{N+F}{F-N}$ and $b = \frac{-2NF}{F-N}$.

Perspective View Volume



The simplest (and most common) perspective transformation defines the view axis to pierce the center of the view volume. We warp the truncated pyramid into the *canonical clipping volume (CCV).*



Canonical Clipping Volume

Mapping to CCV Corners

We project onto the $w \times h$ near plane as follows:

$$\begin{bmatrix} \frac{2N}{w} & 0 & 0 & 0 \\ 0 & \frac{2N}{h} & 0 & 0 \\ 0 & 0 & -\frac{F+N}{F-N} & \frac{-2NF}{F-N} \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{2N}{w}x \\ \frac{2N}{h}y \\ -\frac{F+N}{F-N}z + \frac{-2NF}{F-N} \\ -z \end{bmatrix}.$$

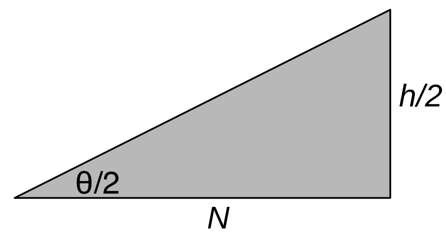
After perspective division we get

$$\begin{aligned} x' &= \frac{-2Nx}{wz} \\ y' &= \frac{-2Ny}{hz}. \end{aligned}$$

Note that the points $(\pm w/2, \pm h/2, -N)$ get mapped to the points $(\pm 1, \pm 1, -1)$. Although its not obvious by inspection, the points $(\pm W/2, \pm W/2, -F)$ get mapped to the points $(\pm 1, \pm 1, +1)$, where W and H are the corresponding dimensions of the far plane.

Field of View and Aspect Ratio

We can solve for the $2N/h$ term based on the vertical *field of view* angle θ. The $2H/h$ term then can be determined from the desired *aspect ratio* $a = w/h$:



$$\begin{aligned} N &= \frac{h}{2}\cot(\theta/2) \\ \frac{2N}{h} &= \cot(\theta/2) \\ \frac{2N}{w} &= \frac{2N}{ah} = \frac{\cot(\theta/2)}{a} \end{aligned}$$

Perspective Projection Transformation

$$P_{\text{persp}}(\theta, \alpha, N, F) = \begin{bmatrix} \cot(\theta/2)/a & 0 & 0 & 0 \\ 0 & \cot(\theta/2) & 0 & 0 \\ 0 & 0 & \frac{F+N}{N-F} & \frac{2NF}{N-F} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

- θ : field of view angle along the vertical axis
  - Large for "fish-eye" lens.
  - Small for "telephoto" lens.
- a : aspect ratio of projection window
  - Often chosen to match the aspect ratio of the viewport (else spheres are warped into ellipsoids).
- $N > 0$: distance to the near clipping plane
- $F > N$: distance to the far clipping plane

```
gluPerspective(fovy, aspect, zNear, zFar)
```

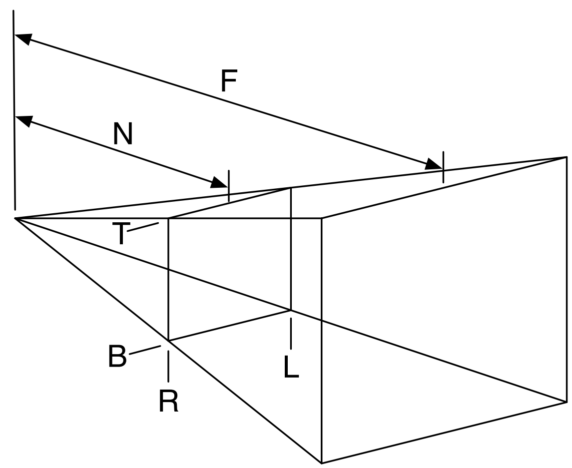
Here is an example that initializes the OpenGL projection matrix with a perspective transformation that uses $\theta = 40^\circ$, $\alpha = 1$, $N =$ hither, and $F =$ yon.

```
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluPerspective(40.0, 1.0, hither, yon);
```

- `zNear` must be strictly positive (not too close to 0).
- `zFar` must be greater than `zNear` but not overly so.
- Wisdom on how to choose these will be revealed when we talk about z-buffering.

Perspective View Frustum

`glFrustum`



A more general specification of a perspective projection transformation (view pyramid not necessarily symmetric about view axis). The near plane becomes the view plane where the boundaries of a rectangle control the shape of the pyramid.

$$P_{\text{persp}}(L,R,B,T,N,F) = \begin{bmatrix} \frac{2N}{R-L} & 0 & 0 & 0 \\ 0 & \frac{2N}{T-B} & 0 & 0 \\ 0 & 0 & -\frac{F+N}{F-N} & \frac{-2NF}{F-N} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$