Object in world coordinates (after modeling transforms)

Transform object from world coordinates to standard camera coordinates

Clip against canonical view frustum

Project using standard camera model

Transform object from world coords to camera coords

Further transform so that frustum is canonical frustum.

Step 1: Translate focal point (PRP) to origin; call this translation \( T \). Since we have PRP in camera coordinates (where we now are), the translation vector is simply negative PRP. In particular, in the very common case where PRP is \((0,0,f)\), the translation vector is \((0,0,-f)\)

Window center is:

\[
\begin{pmatrix}
\frac{u_{\text{max}} + u_{\text{min}}}{2} \\
\frac{v_{\text{max}} + v_{\text{min}}}{2} \\
0
\end{pmatrix}
\]

Window center is now:

\[
\begin{pmatrix}
\frac{u_{\text{max}} + u_{\text{min}}}{2} \\
\frac{v_{\text{max}} + v_{\text{min}}}{2} \\
-1
\end{pmatrix}
\]

1. Translate focal point to origin
2. Shear so that central axis of frustum lies along the z axis
3. Scale \( x, y \) so that faces of frustum lie on conical planes
4. Isotropic scale so that back clipping plane lies at \( z=1 \)

Since we are now in camera coordinates, we will often refer to them as \((x,y,z)\) not \((u,v,n)\).
Step 2: Shear this volume so that the central axis lies on the z-axis. This is a shear, because rectangles on planes $z=constant$ must stay rectangles. Call this shear $S_1$

Shear $S_1$ takes previous window midpoint $\frac{[u_{min}, u_{max}]}{2}$ to $(0, 0, -f)$ - this means that matrix is

\[
\begin{bmatrix}
1 & 0 & \frac{(u_{max} - u_{min})}{2f} & 0 \\
0 & 1 & \frac{(v_{max} - v_{min})}{2f} & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

Note that the size of a rectangle in the image plane does not change.

3. Scale $x, y$ so that planes are on $z=x, z=-x$ and $z=y$ and $z=-y$. Call this scale $S_c$.

4. Isotropic scale so that far clipping plane is $z=1$; call this scale $S_c$.
3. Scale $x$, $y$ so that planes are on $z=0$, $z=x$, and $z=-y$. Call this scale $Sc_3$.

Diagram for $Sc_3$ (ignoring $x$ coordinate)

- $z = f + B$
- Camera plane

$Sc_3 = \begin{bmatrix} \frac{2f}{(v_{\text{max}} - v_{\text{min}})} & 0 & 0 & 0 \\ 0 & \frac{2f}{(v_{\text{max}} - v_{\text{min}})} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

4. Scale $x$, $y$ so that planes are on $z=0$, $z=x$, and $z=-y$. Call this scale $Sc_1$.

\[ \begin{bmatrix} \frac{2f}{(v_{\text{max}} - v_{\text{min}})} & 0 & 0 & 0 \\ 0 & \frac{2f}{(v_{\text{max}} - v_{\text{min}})} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_{\text{max}} \\ v_{\text{min}} \end{bmatrix} = \begin{bmatrix} f \\ f \end{bmatrix} \]

$y = z$

$\frac{1}{2} (v_{\text{max}} - v_{\text{min}}) = f$

$\frac{2f}{(v_{\text{max}} - v_{\text{min}})} = k_y$ (is $y$ scale factor)

5. Now isotropic scale so that far clipping plane is $z=-1$; call this scale $Sc_2$.

Now $y = z$

$Sc_2 = \begin{bmatrix} \frac{2f}{(u_{\text{max}} - u_{\text{min}})} & 0 & 0 & 0 \\ 0 & \frac{2f}{(u_{\text{max}} - u_{\text{min}})} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
5. Now isotropic scale so that far clipping plane is \( z = -1 \); call this scale \( S_c^2 \)

Currently, at far clipping plane, \( z = -f + B \)

Want a factor \( k \) so that \( k(-f + B) = -1 \)

So, \( k = -1 / (-f + B) = 1 / (f - B) \)

(Note that \( B \) is negative, and \( k \) is positive)

\[
S_c^2 = \begin{bmatrix}
1 & 0 & 0 & 0 \\
\frac{1}{f - B} & 0 & 0 & 0 \\
0 & \frac{1}{f - B} & 0 & 0 \\
0 & 0 & \frac{1}{f - B} & 1
\end{bmatrix}
\]

Note that the focal length, \( f \), also gets transformed (needed for the perspective transformation coming up).

It is:

\[
\frac{f}{f - B}
\]

3D Viewing Pipeline

Point in canonical camera coordinates \( S_c^2 S_c^1 T_2 R T_1 \)

Point in world coordinates
Plan A: Clip against canonical frustum (relatively easy—we chose the canonical frustum so that it would be easy!)

Plan B: Be even more clever. Further transform to cube and clip in homogenous coordinates.

Plan A: Clipping against the canonical frustum

2D algorithms are easily extended. For example, for Cohen Sutherland we use the following 6 out codes:

\[
\begin{align*}
& y > z & y < z & x > z & x < z & z < 1 & z > z_{\text{min}} \\
& (x_{\text{min}} = (F-F)/(B-F))
\end{align*}
\]

Recall C.S. for segments

Compute out codes for endpoints

While not trivial accept and not trivial reject:

Clip against a problem edge (one point in, one out)

Compute out codes again

Return appropriate data structure

Clipping against the canonical frustum

Clipping polygons in 3D against canonical frustum planes is simpler and more efficient than the general case.

Recall the S.C. gives four cases:

- Polygon edge crosses clip plane going from out to in
  - emit crossing, next vertex

- Polygon edge crosses clip plane going from in to out
  - emit crossing
  - Polygon edge goes from out to out
  - emit nothing

- Polygon edge goes from in to in
  - emit next vertex

(The above is from before, just change “edge” to “plane”)