We will first map world coordinates to the camera coordinates (top figure).

However, if clipping against the frustum is difficult because planes bounding the frustum have a complex form.

**Solution:** further transform frustum to a canonical form where clip planes are an easy form to deal with:

\[
\begin{align*}
    z &= x, \
    z &= -x, \
    z &= y, \
    z &= -y, \
    z &= -1, \
    z &= d
\end{align*}
\]

\(u\) and \(v\) can be used to specify a window in the image plane; only this section of image plane ends up on the screen.

This window defines four planes; points outside these planes are not rendered.

Hither and yon clipping planes, which are always given in terms of camera coordinates, and always parallel to the film plane, give a volume - known as the view frustum.

Orthographic case: view frustum is cuboid (i.e. all angles right angles, but edges not necessarily of equal length).

If image plane transforms to \(z=m\) then in new frame, projection is easy:

\((x, y, z) \rightarrow (m x/z, m y/z)\)
Step 1. Translate the camera at VRP to the world origin. Call this \( T_1 \).

Translation vector is simply negative VRP.

(We are changing the coordinate system of the world, which is the same thing mathematically as moving the camera. We want object world coordinates to change so that the camera location becomes the origin).

Step 2. Rotate camera coordinate frame (in w.c.) so that so that \( u \) is \( x \), \( v \) is \( y \), and \( n \) is \( z \). The matrix is?

(We are changing the coordinate system of the world, which is the same thing mathematically as moving the camera. We want object world coordinates to change so that the camera axis becomes the standard axis—e.g, \( u \) becomes \((1,0,0)\), \( v \) becomes \((0,1,0)\) and \( n \) becomes \((0,0,1)\)).

In the current coords (world shifted so that VPR is at origin): \( u \) maps into the X-axis unit vector \((1,0,0,0)\) which is what we want.

(Similarly, \( v \)-->Y-axis unit vector, \( n \)-->Z-axis unit vector)
Object in world coordinates (after modeling transforms)

Transform object from world coords 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.

Since we are now in camera coordinates, we will often refer to them as (x,y,z) not (u,v,n).

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

Step 1: Translate focal point (PRP) to origin; call this translation $T_2$. 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:

$$\left(\frac{u_{\text{min}}+u_{\text{max}}}{2} , \frac{v_{\text{min}}+v_{\text{max}}}{2}, 0\right)$$

Window center is now:

$$\left(\frac{1}{2}(u_{\text{min}}+u_{\text{max}}), \frac{1}{2}(v_{\text{min}}+v_{\text{max}}), -f\right)$$
Step 1 is relatively straightforward, but notice that the location of the clipping planes also gets shifted. So, before we had the back clipping plane at B (which is negative). Now it is at: B-f.

Shear $S_1$ takes previous window midpoint $\left(\frac{1}{2}(u_{\text{min}}+u_{\text{max}}), \frac{1}{2}(v_{\text{min}}+v_{\text{max}}), -f\right)$ to (0, 0, -f) - this means that matrix is:

$$
\begin{pmatrix}
1 & 0 & \frac{(u_{\text{min}} + u_{\text{max}})}{2f}
\end{pmatrix}
\begin{pmatrix}
0 & 1 & \frac{(v_{\text{min}} + v_{\text{max}})}{2f}
\end{pmatrix}
\begin{pmatrix}
0 & 0 & 1
\end{pmatrix}
$$

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

Step 2: Shear this volume so that the central axis lies on the z-axis. This is a shear, because rectangles on planes $z=\text{constant}$ must stay rectangles. Call this shear $S_1$.
3. Scale x, y so that planes are on $z=x$, $z=-x$ and $z=y$ and $z=-y$. Call this scale $S_{c1}$.

4. Isotropic scale so that far clipping plane is $z=-1$; call this scale $S_{c2}$.

4. Scale x, y so that planes are on $z=x$, $z=-x$ and $z=y$ and $z=-y$. Call this scale $S_{c1}$.

$$\begin{bmatrix}
\frac{1}{2}(v_{\text{max}} - v_{\text{min}}), & -f \\
\frac{1}{2}(v_{\text{max}} - v_{\text{min}}) & f
\end{bmatrix} \rightarrow (f, -f) \quad \text{(because $y=-z$)}$$

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

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

$$S_{c1} = \begin{bmatrix}
\frac{2f}{(v_{\text{max}} - v_{\text{min}})} & 0 & 0 & 0 \\
0 & \frac{2f}{(v_{\text{max}} - v_{\text{min}})} & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}$$
5. Now isotropic scale so that far clipping plane is \( z = -1 \); call this scale \( \text{Sc}_2 \)

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)

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

It is:

\[
\frac{f'}{f - B} = f
\]
Further comments on the canonical frustum

Note the approximate reciprocal relation of $u_{\text{min}}$, $u_{\text{max}}$, $v_{\text{min}}$, $v_{\text{max}}$, with $f$.

Because of this, the clipping plane values change, and $B_1-f_1 = B_2-f_2$.

Further comments on the canonical frustum

For assignment three, you need to choose $u_{\text{min}}$, $u_{\text{max}}$, $v_{\text{min}}$, $v_{\text{max}}$, and $f$.

I suggest simply setting $u_{\text{min}}$, $u_{\text{max}}$, $v_{\text{min}}$, $v_{\text{max}}$, to reflect your understanding of your screen window in world coordinates, and set $f$ accordingly. (Best to keep the aspect ratio the same).
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.H. 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”)

Plan A: Clipping against the canonical frustum

2D algorithms are easily extended. For line clipping with Cohen Sutherland we use the following 6 out codes:

\[ y > z \quad y < z \quad x > z \quad x < z \quad z < 1 \quad z > z_{\text{min}} \]

\[ z_{\text{min}} = \frac{(F - f)}{(B - f)} \]

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: Clip against canonical frustum

Transform object from world coordinates to standard camera coordinates

Clip against canonical view frustum

Project using standard camera model

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

- For any camera, can turn the view frustrum into a regular parallelepiped (box). We will use the box bounded by \( x = \pm 1, y = \pm 1, z = -1, \) and \( z = 0. \)
- Advantages
  - Simplified clipping in homogenous coordinates
  - Extends to cases where we use homogenous coordinates to represent additional information (and \( w \) could be negative).
  - Can simplify visibility algorithms.
- Approach: clever use of homogenous coordinates

Transforming canonical frustum to box

Do this in two steps. One stretch in \( y \) (and \( x \)), and on stretch in \( z \).
Transforming canonical frustum to box

The picture should suggest an appropriate scaling for $y$. It is $y' = \frac{y}{-z}$.

On top, $y \rightarrow 1$, so scaling is $\frac{1}{y}$. Recall that $y=z$ there.

On bottom, $y \rightarrow -1$ so scaling is $-\frac{1}{y}$. Recall that $y=z$ there.

So scaling is $y' = \frac{y}{-z}$.

Similarly, $x' = \frac{x}{-z}$.

Transformation is non-linear, but in h.c., we can make $w = -z$.

For $z$, we translate near plane to origin. But now box is too small. Specifically it has $z$ dimension $\frac{1}{1 + z_{\text{min}}}$ (recall $z_{\text{min}}$ is negative).

So we have an extra scale factor $\frac{1}{1 + z_{\text{min}}}$ and thus $z' = \frac{(z - z_{\text{min}})}{(1 + z_{\text{min}})}$.

But we want $x$ and $y$ to work nicely in h.c., with $w=-z$, so we use $z' = \frac{(z - z_{\text{min}})}{1 + z_{\text{min}}} / (-z)$

(Thus in our box, depth transforms non-linearly)

In h.c.,

$x \rightarrow x$

$y \rightarrow y$

$z \rightarrow \frac{(z - z_{\text{min}})}{(1 + z_{\text{min}})}$

$l \rightarrow -z$

So, the matrix is
In h.c.,
\[ x \Rightarrow x \]
\[ y \Rightarrow y \]
\[ z \Rightarrow (z - z_{\text{min}}) / (1 + z_{\text{min}}) \]
\[ 1 \Rightarrow -z \]

So, the matrix is
\[
\begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 + z_{\text{min}} & \frac{-z_{\text{min}}}{1 + z_{\text{min}}} \\
0 & 0 & -1 & 0
\end{pmatrix}
\]

Mapping to standard view volume (additional comments)

- The mapping from \([z_{\text{min}}, -1]\) to \([0,-1]\) is non-linear. (Of course, there exists a linear mapping, but not if we want everything else to work out nicely in h.c.).
- So a change in depth of \(\triangle D\) at the near plane maps to a larger depth difference in screen coordinates than the same \(\triangle D\) at the far plane.
- But order is preserved (important!); the function is monotonic (proof?).
- And lines are still lines (proof?) and planes are still planes (important!).

Clipping in homogeneous coordinates

- We have a cube in \((x,y,z)\), but it is not a cube in homogeneous coordinates, so we must divide if we want to take advantage of this particularly nice clipping situation.
- However, dividing before clipping might be inefficient if many points are excluded, so we often clip in homogeneous coordinates.
Clipping in homogeneous coord.’s

- Write h.c.’s in caps, ordinary coords in lowercase.
- Consider case of clipping stuff where x>1, x<-1
- Rearrange clipping inequalities:

\[
\begin{align*}
\frac{X}{W} > 1 & \quad \Rightarrow \quad X > W, \\
\frac{X}{W} < -1 & \quad \Rightarrow \quad X < -W, \\
W > 0 & \quad \text{AND} \quad X < W, \\
W < 0 & \quad X > -W,
\end{align*}
\]

(For now W has been positive, but negatives occur if we further overload the use of h.c.’s)

Reminder of the last steps

In both plans we need to project into 2D.

If we are working in the canonical view space, then we project using the standard camera model (easy) and divide.

Recall that the matrix for the standard camera model using homogeneous coordinates is:

\[
\begin{pmatrix}
1 \\
1 \\
1 \\
1/f \\
0
\end{pmatrix}
\]
Reminder of the last steps

If we are working in homogenous coordinates, then we first divide and then projection is even easier (ignore z coordinate).

The mapping to the box—which was complete once the division was done—implicitly did the perspective projection—essentially we transformed the world so that orthographic projections holds.

Finally, we may need to do additional 2D transformations.

In the canonical frustum case, our (x,y) coordinates are relative to (-f',f'). They need to be mapped to the viewport (possibly implicitly by the graphics package).

In the canonical box case, our (x,y) coordinates are relative to (-1,1). They need to be mapped to the viewport (possibly implicitly by the graphics package).